AMIGA NEWS: CDTV SEE PAGE 47

Amazing COMPUTING™ / AMIGA

For The Common

Your Original AMIGA® Monthly Resource

Volume 5 No. 7 July 1990
US $3.95 Canada $4.95

SPEED TRIALS:

AMIGA A2000 ACCELERATORS

PART ONE OF A COMPARATIVE ANALYSIS

PLUS!

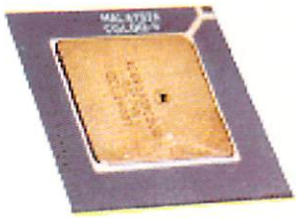
FIXING A LIGHTNING BUG ON COMMODORE MONITORS

CUSTOM POINTERS IN BASIC

PIXOUND AND HYPERCHORD

07
0 74470 74710 9

CONTENTS



Motorola 68030 microprocessor. **See article entitled *Apples, oranges and MIPS* on 68030 accelerators.**

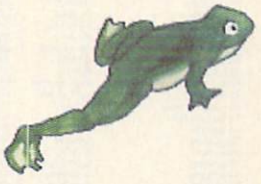
INSIDE

REVIEWS

COLUMNS

Amazing COMPUTING™ For The Commodore AMIGA
Your Original AMIGA® Monthly Resource

Vol. 5
No. 7
July, 1990



Cover by
Ernest P. Viveiros, Sr.

AMAZING DEALERS

The following **Amazing Dealers**, carry *Amazing Computing*™, your resource for information on the Amiga™, and *AC's Guide To The Commodore Amiga*, the total Amiga product guide. **1-508-678-4200**
If you are not an Amazing Dealer, but would like to become one, call.

Alaska
Best Electronics — Anchorage
Juneau Electronics — Juneau
Alabama
Alabama Computers Inc. — Huntsville
Madison Book & Computer — Madison
Pasco Industries — Birmingham
The Computer Image — Birmingham
The Computer Nut — Birmingham
Universal Computer System — Mobile
Arkansas
Commodore Connection — Sherwood
Computer Concepts — Gosnell
Microtronix — Fort Smith
The Micro Shop — Little Rock
Arizona
Bookman's — Tucson
Books Brothers Ltd. — Tucson
Books Etc. — Tempe
Brain Child Software — Phoenix
Computer Library, Inc. — Tempe
Livingston's Books, Inc. — Sierra Vista
North American Digital — Tucson
Try Tech Inc. — Phoenix
California
3 E Software And Systems — Hayward
Adav International — Foster City
American Mags & Books Ltd. — Englewood
Arcadia Newsland — Arcadia
B. Dalton Books — San Rafael
Balboa Office Supply — San Diego
Bits & Bytes — Fairfield
Bits n Pieces Comp Prod. — Anaheim
Bookland — Marysville
Bookstar — San Diego
Brentano's — Sacramento
Brentano's — San Francisco
Brown Knows Computing — Redlands
CAL-Tech Bookstore — Pasadena
Candy Computer — Elk Grove
Castro Kiosk — San Francisco
Catri Product Company — Burlingame
Centerfold News — Los Angeles
Central Park Bookstore — San Mateo
Century Computer Systems — La Habra
Circus of Books — Los Angeles
Circus of Books — Los Angeles
Commodore Land — Alhambra
Commitech — National City
Computer Attic — Palo Alto
Computer Literacy Books — Sunnyvale
Computer Literacy — San Jose
Computer Literacy — Santa Clara
Computer Nook, Inc. — San Bernadino
Computer Outlet — Chula Vista
Computer Outlet — San Marco
Computer Room — Scotts Valley
Computer Showcase — San Francisco
Computer Spectrum — Burlingame
Computer Time — Citrus Hts
Computers Plus — Grove City
Coronet Newsstand — Oceanside
Covina Bookstore — Covina
Dataphile — Watsonville
Del Amo Books & News — Torrance
Delaurer's News Agency — Oakland
Discount Computer Station Inc. — Long Beach
ECX Computer Co. — Walnut Creek
Empire Computers — Santa Rosa
Encinitas News — Encinitas
Fil-American — San Francisco
Fisher Computers + Software — Rancho Cucamonga
Harding Way News — Stockton
Heinrichson, Schneider & Young, Inc. — Berkeley
Home Computing Center — Concord
Home Computing Center — Mill Valley
Home Computing Center — San Leandro
HT Electronics — Sunnyvale
Int's Periodical Dist — San Diego
International Telecine — Santa Ana
K J Computers — Granada Hills
L. A. Int's News — Venice
Little Professor Books — Placentia
Lyon Enterprises — Berkeley
Lyon Enterprises — Berkeley
Micro P. Technologies — Lawndale
NewsBeat — Davis
Norma McCay — San Diego
Northridge Newsstand — Northridge
Op Amp Tech Books — Los Angeles
Paras Newsstand — San Diego
Parsin International — San Diego
Printers Inc. — Palo Alto
Pul's Electronics — North Highlands
R&R Software — Chula Vista
Ridgecrest Computer Ctr, Inc. — Ridgecrest
S.O.S. Computers — Los Angeles
Sawyer's News Inc. — Santa Rosa
Siskiyou Business Systems — Mount Shasta
Software Cellar — Anaheim
Software Central — Pasadena
Software First — Santa Rosa
Software First — Vallejo
Software Plus — Citrus Heights
Software Solutions Plus — El Cerrito
Stacy's #5 — Modesto
Technitronics Service — South Pasadena
The Booksmith — San Francisco
The Lively Computer — El Cajone
Third Avenue News — Chula Vista
Tower Books — Chico
Tower Books — Citrus Hts.
Tower Books — Concord
Tower Books — Mountain View
Tower Books — Sacramento
Tower Books — San Mateo
Universal News — Hollywood
Vintage Wine — San Diego
Warner Engineering — San Diego
WCI Computer Central — Eureka
Winners Circle — Berkeley
World Book & News — Hollywood
Colorado
Aurora Newsstand — Aurora
Citadel Computers — Colorado Springs
Computer Discount — Denver
Computer Repeats — Boulder
Computer Room — Aurora
Micro World — Lakewood
Paragon Computers — Colorado Springs
Paragon Software & Electronics — Colorado Springs
Sun Country Computers — Colorado Springs
Tattered Cover Bookstore — Denver
Westminister Newsland — Westminister
Connecticut
Aetna Life Club Store — Hartford
Amiga Business Computer — Norwalk
Beta Press — Derby
Computer & Software Connections — Plantsville
Derric Electronics — Hamden
Friendly Computers — Southington
G & G Christ Church — Derby
Gordon & Gotch — Inlectronics, Inc.
Int'l Mag Dist/Capetown — Woodbury
Lac Distributors — Derby
Lebanon/Levant Dist. — Norwalk
Libery Book — Quaker Hill
Mnemonics Inc. — Orange
One Byte — East Hartford
Software City — North Windham
Software Kingdom
Tricom Computers
District of Columbia
Programa — Washington
World News — Washington
Delaware
Newark Newsstand — Newark
Florida
Am Comp Computer Center — Oviedo
Bookstop — Clearwater
Bookstop — Miami
Bookstop — Orlando
Brentano's — N. Palm Beach
Chuck's Newsstand — Vera Beach
City Newsstand — Melbourne
Commodore Country Inc. — Pinellas Park
Computer Bar — Pensacola
Computer Base — Brandon
Computer Image — Miami
Computer Lane Inc — Pembroke Pines
Computer Room — Ft. Lauderdale
Computer Specialties — Melbourne
Computers + Video Creations — Palm Beach
Computers Etc. — Sarasota
Computers Plus — S. Daytona
Creative Equipment — Miami
Extra Extra — Tampa
Florida Book Store — Gainesville
Gen-Rod — Miami
Gulf Coast Computer Exchange — PanamaCity
Joe's News, Inc. — Miami
Malvar Freight — Miami
Mark Twain Smoke Shop — Lakeland
Micro Computer Center — Largo
New Age Electronics — Clearwater
New Age Electronics — St. Petersburg
PC Collection — Coral Gabels
Program Store — Clearwater
Program Store — Pinellas Park
Prospect News & Books — Ft. Lauderdale
Public Domain Solutions — Sarasota
Software City — Orange Park
Sunny's At Sunrise, Inc. — Sunrise
Sunshine Software — Ft. Lauderdale
Georgia
Arpet System — Norcross
BFJ Enterprises — Roswel
Brentano's — Atlanta
Cardinal Computers — Dalton
Micro Center/Marietta — Marietta
Mostly Software — Columbus
Oxford Books, Inc. — Atlanta
Software City Savannah — Savannah
Software South Inc. — Savannah
Techno World — St. Mary's
The 64 Store — Atlanta
Hawaii
Astro Computers & Electronics — Aiea
Computronics — Honolulu
Data-One Computer Center — Honolulu
J & C Repair — Aiea
Idaho
A.B.I. Computer & Video — Nampa
The Creative Edge — East Ketchum
Illinois
Audio Visual Rec. — Chicago
Books & Bytes, Inc. — Naperville
Computer Software Connection — Moline
Comp—Kingdom — Lisle
Coopersmith — Vernon Hills
Digital World — Naperville
DigitalWorld — Addison
DOS Computer Center — Niles
FD Software & Caddis — Bloomingdale
Keep In Pace Computers — Champaign
Krochi's & Brentano's — Aurora
Krochi's & Brentano's — Chicago
Krochi's & Brentano's — Orland Park
Krochi's & Brentano's — Schaumburg
MZJ2 Inc. — Champaign
Megabyte Computer Works — Springfield
Memory Expansion — Bloomington
Micro Age — Oaklawn
Micro Ed Enterprises Inc. — Chicago
Micro Tech Soultions — Aurora
Micro Tek — Kankakee
Ring Software — Geneva
Software City — Belleville
Software Plus Chicago — Chicago
Software Plus West — Hanover Park
Sofware Plus — Wheeling
Unique Computer — Sterling
Warren Radio/FRANCO — Peoria
Indiana
Bookland — Indianoplis
Burkat Computer Ctr. — South Bend
Cal Com Inc. — Kokomo
Computer Corner — Ft. Wayne
Computer Corner — Marion
Computer People Inc. — Michigan City
CPU Inc. — Indianapolis
Indiana News — Indianapolis
Micro Computer Centers Inc. — Indianapolis
Programs to Go — Muncie
The Book Corner — Bloomington
Vons Computers — West Lafayette
Xerox Computer Center — Indianapolis
Iowa
Computer Solutions & Facts — Des Moines
Computers & More — Des Moine
Gronert Computers — Ames
Soft Search
Kansas
Arc Electronic Distributors — Wichita
Brentanos — Mission
Micro Medis — Hays
Midwest Computers — Manhattan
Software City — Overland Park
Software Exchange of KS — Wichita
Kentucky
Computer Dimensions Inc. — Lexington
Computer Mania Inc. — Florence
Expert Services — Louisville
Hawley Cooke — Louisville
Hawley Cooke — Louisville
Reality World Computers — Lexington
Soltech Computer Systems — Lexington
The Computer Lab — Louisville
Louisiana
Bookstar — New Orleans
City Newsstand — Baton Rouge
Computer Time — Shreveport
Lakeside News — Metairie
Modern Business Machines — Metairie
Software Center International — Metairie
Software Source — Lafayette
The Book Shelf — Slidell
Maine
Hands On Computing — Westbrook
Silylight Software — Belfast
Maryland
Buried Treasure — Rockville
Capital Classics Ltd — Silver Spring
Greetings and Readings — Towson
Knowledgeware Systems — Kensington
Micro Computer Center — Baltimore
New Age Computers — College Park
Professional Micro Services — Baltimore
Software Advantage — Rockville
Software City — Rockville
Massachusetts
Computer Works — Middletown
Computer Zone — North Attleboro
Dox — Williamstown
HCS Computer Center — Marshfield
LCA Video & Computer Ctr. — Norwood
Memory Location — Wellsley
Newsbreak, Inc. — Swansea
OmniTek Computers — Tewksbury
Pioneer Valley Data Equipment — Amherst
Soft Designs — Belmont
Softpro — Burlington
Software Connections — Swansea
The Bit Bucket — West Newton
Tycom Inc. — Pittsfield
Valdata — Hadley
Michigan
Book Center Of E. Detroit — E. Detroit
Borders Book Shop — Novi
Central Park Computers — Okemos
CompuSoft — Kalamazoo
Computer Emporium — Lansing
Coopersmith's — Livonia
Direct Access — Lansing
Global Computer — Dearborn
I/O Software — Flint
Little Professor Books — Brighton
Michigan News Agency — Kalamazoo
Michigan Software — Novi
Micro World — Livonia
Micro World — Milford
Micro World — Roseville
Micro World — Southfield
Micro World — Southgate
Micro World — Sterling Heights
Microworld Computer Center — Dearborn
Midland Computer Shop — Midland
Midwest Micro Computer — Lapeer
New Horizons Bookshop — Roseville
Norman Business Control Sys. — Wyoming
Readmore Newscenter — Lansing
Rochester Book Center — Rochester Hills
Missouri
Assoc Computer Services — Springfield
Brands Mart Computers — Kansas City
Clark Worldwide — Kansas City
Data Byte Computer Co. — St. Louis
Software City — Independence
Systems Plus of St. Louis — St. Louis
Systems Plus — Overland Park
Nebraska
My Com Inc. — Plattsmouth
Double E Computer Systems — Omaha
Nevada
Computer House — Sparks
Nat'l Discount Computer — Las Vegas
Tower Records — Las Vegas
New Hamoshire
Dartmouth Bookstore — Hanover
Digital Connection Computers — Dover
Diversified Computers — Keene
System Eyes — Nashua
New Jersey
Act One — Vineland
Atlantic News — Atlantic City
Business System Professional — Scotch Plains
Connecting Point — Union
Enschpea Enterprises — Fort Lee
Family Computer Centres — South Orange
Garden State News — Cliffside Park
Hudson News — New York
Morris County Stationers — Chester
Program Store — Princeton
Software Station — Rockaway
The Program Store — Eston Town
The Program Store — Wayne
Unlimited Software Inc. — Rutherford
Village Computer Center — Cedar Knolls
Waldensoftware — Paramus
Wilson Group USA — Jersey City
New Mexico
New Horizon Computer System — Alamogordo
Newsland Bookstore — Albuquerque
Page One Newsland — Albuquerque
Road Runner — Santa Fe
New York
Amagination — New York
Amcorp Ltd — Huntington Station
Amicus- Hudson Valley — Wappingers Falls
Amigo Business Computers — East Northport
B.C. Communications — Huntington Station
Book Distributor — Huntington Station
Broadway News — Saratoga Springs
Byte Shop — Merrick
Compunite Center — White Plains
Computer Metropolis — New Paltz
Computer Outlet — Jamestown
Contemporary Computing — Latham
Creative Computers — Binghampton
Cromland Inc. — Brooklyn
Global Software — New Hyde Park
Island Software Computer Services — Coram
Kemco Publisher Ser — East Northport
Leigh's Computers — New York
MicroWorks — Kenmore
Mnemonics — North Port
Oxide Industries Inc. — Westbury
QB Software, Inc. — Forest Hills
Readers Market — Tonawanda
Rochester Books — Rochester
Sherry Tunkel — New York
Singapore Book — Huntington Station
Software & Such — Scotia
Software City — Spring Valley
Software House — Fairport
Software Link — White Plains
Software Supermarket II — Cheektowaga
Syntaire Trading — New York
Sysput Computers — Brooklyn
The Computer Cellar — Albany
The MicroWorks — Tonawanda
Unicom Electronics — Johnson City
Video Computer Bin — Brooklyn
Video Computer Center — Rome
Village Computer Inc. — New York
World Computers Inc. — Hicksville
North Carolina
Digitz — Raleigh
K&S Newsstand — Clemmons
K&S Newsstand — Kernersville
K&S Newsstand — Winston-Salem
News Center — Cary
Parker News Inc. — High Point
Servatronic Technologie — Burlington
Software City — Charlotte
Software City — Fayetteville
Software City — Raleigh
North Dakota
Computer Associates — Fargo
G. F. Computers — Grand Forks
Software House — Fargo
The Computer Store — Minot
Ohio
BS Software — Columbus
Brentanos — Cincinnati
Cincinnati Fin SQ News — Cincinnati
Computer Network — Newark
Eddy's Cigar Store — Elyra
Fairborn Home Computer — Dayton
Infinity Computing — Columbus
Lakewood International News — Lakewood
Leo's Book & Wine Shop — Toledo
Local Computer Store — Chesterland
Medical Computers — Centerville
Micro Center-Columbus — Columbus
Micro Center-Westerville — Westerville
Microtech Computer Center Inc. — Toledo
Microware Magic — Fairfield
News Shop — Kent
North Coast Programming — Willoughby
Not Just Software — South Toledo
Not Just Software — West Toledo
Paper & Ink — Tiffin
PC Powerhouse, The — Fairfield
Power Computing Inc. — Toledo
Salem Computer Center — Salem
Software Centre International — North Olmsted
Software and More — Cincinnati
Software Connection — Mayfield Hts.
Somers Photo & Comp. — Cleveland Heights
Upgrade — North Olmstead
Oklahoma
Little Professor Books — Stillwater
Merit Computers — Oaklahoma City
Micro Link Computer Store — Tulsa
Second Hand Software — Oklahoma City
Video-Comp Inc. — Lawton
Oregon
Andromeda — LaGrande
Clackamas Computers — Beaverton
Clackamas Computers — Clackamas
Comm-Shack — Salem
Computer U.S.E.R.S. — Springfield
Libra Books Inc. — Eugene
News & Smoke Shop MF — Medford
PC Express — Eugene
Powell's Tech — Portland
Rich Cigar Store — Portland
Sath & Washington News — Portland
Soft Cell Software, Inc. — Portland
Software Pipeline-Salem — Salem
Software Pipline — Beaverton
Tech Books — Portland
Pennsylvania
ATD Software — Sayre
Computer Basics Inc. — Erie
Bryn Mawk News — Coles, Book People
CDR Systems — Pittsburgh
Circle Computer — Ephrata
Clark Worldwide — Middletown
Computer Basics Inc. — Hermitage
Computer Wurx — Stroudsburg
Computers and Games — Reading
Elmo's Computers — Brownsville
Floppy House Software — Palmyra
Gene's Books — King of Prussia
Graphic Computer Center — Lionville
Lauer-Todd Computer Solutions — Newtown
Micro Master Inc. — Monaca
North Central Digital Systems — Danville
Pittsburgh Computer Store — Pittsburgh
Software Hut Inc. — Philadelphia
Some Hole In The Wall — Philadelphia
T A Slick & Co, Inc — Lancaster
Tasco Computer Center — Lancaster
The Floppy Disk — Mechanicsburg
Triangle Computers — Indiana
Walden Hill Images — Berlin
Rhode Island
Micro Limits — Warwick
Software Connections — Warwick
The Computer Place — Woonsocket
South Carolina
Avcom International Inc. — West Columbia
Software Haus — Charleston Hts
Watson's Jewlery — Chesterfield
Tennessee
Bristol Micro Technolog — Bristol
ComputerLab of Memphis — Memphis
Computer Hardware Services — Nashville
T.Y. Corp — Bartlett
Texas
A&E Computers — Grove
All Systems Go Software — San Antonio
Amazing Computer Systems — Dallas
Amazing Computer Systems — Fort Worth
Bookstop — Arlington
Bookstop — Austin
Bookstop — Dallas
Bookstop — Houston
Bookstop — San Antonio
Brentano's — Dallas
Byte Shop — Austin
Computer Ease — Corpus Christi
Computer Magic — Austin
Computer Skills — Eules
FNS Computer Services — Texarkana
Maxwell Book Inc. — Desota
Metropolitan Computer — Richardson
Micro Search — Houston
Software Library — Wichita Falls
Software Terminal — Fort Worth
The Computer Experience — San Antonio
The Computer Stop! — Amarillo
The Federated Group — Mesquite
Trinity News Company — Fort Worth
Utah
Armadillo Bros — Salt Lake City
Magazines and Gifts — Salt Lake City
People's Computers — Provo
Virginia
Audio Fidelity Corp — Richmond
Brentanos — Alexandria
Discover — Fairfax
Family Computer Center — Fairfax
Family Video & Computer Center — Newport News
I/O Computer — Norfolk
Tract Software — Prince William
Video Computer Retailers — Chesapeak
Virginia Micro Systems — Woodbridge
Washington
Amazing Electronics — Spokane
Apple Valley Computers — E. Wenatchee
Bits, Bytes and Nibbles — Spokane
Brentano's — Seattle
Bull Dog News — Seattle
Butlers Computer Services — Federal Way
Byte World — Kent
Computers + Inc. — Tacoma
Family Computers — Everett
Micro Age Computer Store — Bellevue
Nybbles & Bytes — Tacoma
Omni International — Seattle
Port Book and News — Port Angeles
Steve's Broadway News — Seattle
Swan Computers — Yakima
Take A Byte — Kennewick
Tech Star — Kent
The Computer Shoppe — Vancouver
Tower Books — Bellevue
Tower Books — Seattle
Tower Books — Tacoma
Y Not Magazines and Books — Federal Way
Wisconsin
20th Century Books — Madison
Computer Software Center — Milwaukee
Fox Valley Personal Computer — Oshkosh
Madison Programmer's Guide — Madison
Software City — Racine
The Floppy Shop — Janesville
TMW Software Inc. — Wausau
Wyoming
Byte Emporium — Laramie
Computer Works — Laramie
Crazy Mountain Electronics — Casper

Outside of the United States

Canada
Alberta
Coles, Book People — Calgary
Coles, Book People — Edmonton
Software Supermart — Edmonton,
British Columbia
Computime Computer & Software — Prince George
Strider Computer Centres — Delta
Unihall — Vancouver
Manitoba
Adventure Software, Ltd. — Winnepeg
New Brunswick
Computer World — Moncton
P.G. Compusystems Inc. — Campbellton
New Foundland
J.W. Allan Co. Ltd. — St. John's
Nova Scotia
Atlantic News — Halifax
Coles, Book People — Yarmouth
Kobetek Systems — Halifax
Renmatt Enter Ltd. — Greenwood
Ontario
All Computers — Gloucester
Arkon Electronics Ltd. — Toronto
Brittons Smoke Shop — Ottawa
Canadian Software Source — Downsview
Chatham Computer Sales — Chatham
Classic Computer — London
Coffey's News Depot — St. Thomas
Coles Bookstore — Rexdale
Coles, Book People — Quelph
Coles, Book People — Scarborough
Coles, Book People — Toronto
Coles, Book People — Willowdale
Comput-or Software — Barrie
Computer & You, The — Islington
Computer Express — Mississauga
Computer Force — Sudbury
Computer House — Chatham
Computer Odyssey, Inc. — Hamilton
ComputerLand — Cornwall
Computronika — Brampton
Computronics Data Centre — Toronto
Comspec — Toronto
Con-pute — Oshawa
Cryston Computer System — Midland
CVF Data Systems — Peterborough
Daily Planet, The — Whitby
Data Marketing Kingston — Kingston
Desktop Computer, Inc. — Kitchener
Electronics 2001 — Willowdale
Foreign Correspondence — Kingston
Globe Magazine — Ottawa
Gran Bookstores, The — Toronto
Granger T.V. Ltd. — Goderich
Hi-Land News — Barrie
Horizon Computers — Whitby
Humber College Stores — Rexdale
Kahle Electronics — Kanata
Lichtman's — Thornhill
Lichtman's — Willowdale
Lightman's News & Books — Toronto
Litchtman's — Toronto
Lynx Technical Services — Ottawa
Mega & Pags — Ottawa
Mikadon Computer Engineering — Thunder Bay
Mr. Diskettes — Ottawa
Novatech — Niagra Falls
Nu-Info Teck — Ottawa
Peach Micro — Ottawa
Personal Computers — London
Personal Keyboard Systems — London
Prospero Books Ltd. — Ottawa
Smoker's Corner — Pickering
Smoke's Place — Colborne
Smokers Plus — St. Catherines
The Software Project — Hamilton
U of W Bookstore — Waterloo
World's Biggest 200 — Toronto
Zoez Software — Brantford
Zoez Software — Milton
Quebec
Azimut Infor, Inc. — Quebec
Camelot-Info — Montreal
E. M. Press & Books, Inc. — Montreal,
Jupiter Technologies Inc. — Pt-Claire
Logiciel International — Montreal
Multimicro — Ville D'Anjou
Ordinateurs 2000 Inc. — Hull
Quebits Logicel Inc. — Montreal
Star Tek Infosolutions — St. Leonard
Saskatchewan
CD's Computer Connection — Saskatoon,

Distributed in Canada by
Ingram Software Distributors — Concord
Compulit Distributors — British Columbia

Argentina
Logic Computer — Buenos Aires
Australia
Computermart — Leederville,WA
Dialworks — Greenwich, NSW
John Verhoeven — Cloverdale, WA
Maxwell Office Equipment — Abbotsford, Melburn
Run Soft — Melbourne
Saudia Arabia
Abdulla Fouad & Sons — Daman
Scotland
The Amiga Center of Scotland — Edinborough
Sweden
Procomp

AMIGA Dealers, Don't Miss Out! Call for *Amazing Computing* Today!

Amazing Computing™ is also available in most B. Dalton Booksellers, B. Dalton Software Stores, Crown Books, Software Etc., selected WaldenBooks Stores, and Walden's Software Store locations.

Amazing COMPUTING / AMIGA

Amazing Computing For The Commodore AMIGA™

ADMINISTRATION

Publisher:	Joyce Hicks
Assistant Publisher:	Robert J. Hicks
Admin. Assistant:	Alisa Hammond
Circulation Manager:	Doris Gamble
Asst. Circulation:	Brigitte Renee Plante
Corporate Trainer:	Virginia Terry Hicks
Traffic Manager:	Robert Gamble
International Coordinator:	Donna Viveiros
Marketing Manager:	Ernest P. Viveiros Sr.
Programming Artist:	E. Paul

EDITORIAL

Managing Editor:	Don Hicks
Associate Editor:	Elizabeth Fedorzyn
Hardware Editor:	Ernest P. Viveiros Sr.
Technical Editor:	J. Michael Morrison
Video Consultant:	Frank Mc Mahon
Copy Editor:	Aimée B. Abren
Copy Editor:	Lisa Friedlander
Copy Editor:	Andy Patrizio
Copy Editor:	Greg Young
Art Director:	William Fries
Photographer:	Paul Michael
Illustrator:	Brian Fox
Research & Editorial Support:	Marilyn Gagne
Production Assistant:	Melissa-Mae Viveiros

ADVERTISING SALES

Advertising Manager: Donna Marie

1-508-678-4200
1-800-345-3360
FAX 1-508-675-6002

SPECIAL THANKS TO:
Buddy Terrell & Byrd Press
Bob at Riverside Art, Ltd.
Swansea One Hour Photo
Pride Offset, Warwick, RI
Mach 1 Photo

Amazing Computing™ (**ISSN 0886-9480**) is published monthly by PiM Publications, Inc., Currant Road, P.O. Box 869, Fall River, MA 02722-0869.

Subscriptions in the U.S., 12 issues for **$28.00**; in Canada & Mexico surface, **$36.00**; foreign surface for **$44.00**.

Second-Class Postage paid at Fall River, MA 02722 and additional mailing offices.

POSTMASTER: Send address changes to PiM Publications Inc., P.O. Box 869, Fall River, MA 02722-0869. Printed in the U.S.A.

First Class or Air Mail rates available upon request. PiM Publications, Inc. maintains the right to refuse any advertising.

Pim Publications Inc. is not obligated to return unsolicited materials. All requested returns must be received with a Self Addressed Stamped Mailer.

Send article submissions in both manuscript and disk format with your name, address, telephone, and Social Security Number on each to the Associate Editor. Requests for Author's Guides should be directed to the address listed above.

EDITORIAL CONTENT

Commodore Announces CDTV

THE CONSUMER ELECTRONICS SHOW (CES) held twice each year, has always been regarded as the first place to see and experience the consumer products that will shape our lives. At CES everything from FAX phones to car stereos are shown to the large number of store buyers, investors, and press who gather every six months to see what direction the world is taking.

Commodore Business Machines usually attends CES, they have always maintained products such as the C64 and 128 computers as consumer items. It was a little unusual, however, that CBM did not attend Comdex (running on almost the exact same dates in Atlanta). Whispers, rumors, and wild guesses all suggested that something major was in the works.

Surrounded by Nintendo, Sega, and a host of entertainment software companies, Commodore's booth demonstrated Amiga 3000's, A2000's, A500's, and even a few very special MS-DOS machines (including a slick little laptop, the 286LT). Yet, still further in the booth, behind closed doors, stood the newest of Commodore's products. A small black box, extremely similar in appearance to a VCR, was performing Amiga software and demonstrations from compact discs. While booth personnel were busy demonstrating the separate abilities of the new Commodore Interactive Graphics Player, they repeatedly reminded everyone that the player was not a computer. It was a new education/entertainment appliance.

Commodore calls this new format Commodore Dynamic Total Vision or CDTV. Amiga developers call it opportunity.

Hybrid Technology

Once again, Commodore has been able to set a standard and create a technology before its competition. Compact disc players for computers is not a new concept. However, this hybrid technology of an Amiga computer integrated with a compact disc player permits the user access to vast amounts of computer ability without using a computer.

The technique is similar to that found in everything from refrigerators to automobiles. Computers are now working quietly in the kitchens and under the hoods of a major portion of the American population. This has been successful because the computer is integrated into the design of each unit and the consumer is never face to face with the computer. People who would balk at using a keyboard, have no difficulty setting the timer on their microwave or starting their car. Yet in each incident, these people have interfaced with a dedicated computer and have instructed that computer to perform a function.

Commodore has taken this concept and integrated the multitasking and special graphics capabilities of the Amiga with the large format and (almost) indestructible nature of the compact disc. Commodore views the CDTV player as a means for any person to research a subject, entertain friends, or present a business proposal with computer efficiency without the need to access the computer.

Although Commodore executives have flatly stated that they will not pre-announce products, they have quietly suggested that peripherals for existing Amiga 3000's, 2000's, and 500's would not be far behind. Their interest in making the CDTV format available to Amiga owners is to tap the already large installed base of Amigas.

CDTV Future

With CDTV, we are witnessing the first views of what life will be like for all of us in the next decade. If we can forecast the future based on the directions of the present, then we must assume that the age of information will continue. We are forced to view the abilities of video as more than a means of entertainment and look at the advances the Amiga has brought to the media in presentations with graphics and sound. Video will continue to become more integrated in our lives and our work.

The Commodore Interactive Graphics Player is the first tool designed to take advantage of both computer ability and ease of use. One of the main hurdles in using computers more successfully in education has been the inability to orchestrate the computer into the general curriculum. Computers became an extra subject instead of a means to teach existing ones. Slowly small steps have been made to integrate the computer into subject matter, but the advances have remained minimal and far from the mainstream.

Now Commodore has offered the educators of our world the opportunity to address vast amounts of material and integrate it into a presentation. Whether it involves researching items through an encyclopedia or discovering the world through an interactive atlas, teachers now have a tool that is easier to use than a film projector, and one that is a lot more predictable. Students can become involved in their studies instead of being passive. With the right software, CDTV can become the one format that no school can afford to be without.

Home CDTV

The same features that make CDTV so important in an academic environment, will also make it a valuable tool at home. Yes, there are people who have spent thousands of dollars for encyclopedias, globes and reference books, only to have their children avoid them like the plague. However, CDTV is different.

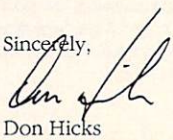
CDTV in the home can be an educator that works as an entertainer. The entertainment market is very excited about the possibilities of what is available through CDTV. One of the most often asked questions at CES in the entertainment booths were "Have you seen Commodore's new player and what did you think of it." This was almost always followed by the statement, "We will be doing something for it real soon."

If Commodore's third party vendors come through, there will be 100 titles for the CDTV player by the launch in September with an additional 100 titles available by the holidays. Add the optional Amiga disk drive and the CDTV player will be able to access all the software currently available for the Amiga. One caveat, the software on the CDTV player will only be as useful as the peripherals you have purchased: remember, if a program calls for keyboard input and you have no keyboard, you are stopped.

Innovation

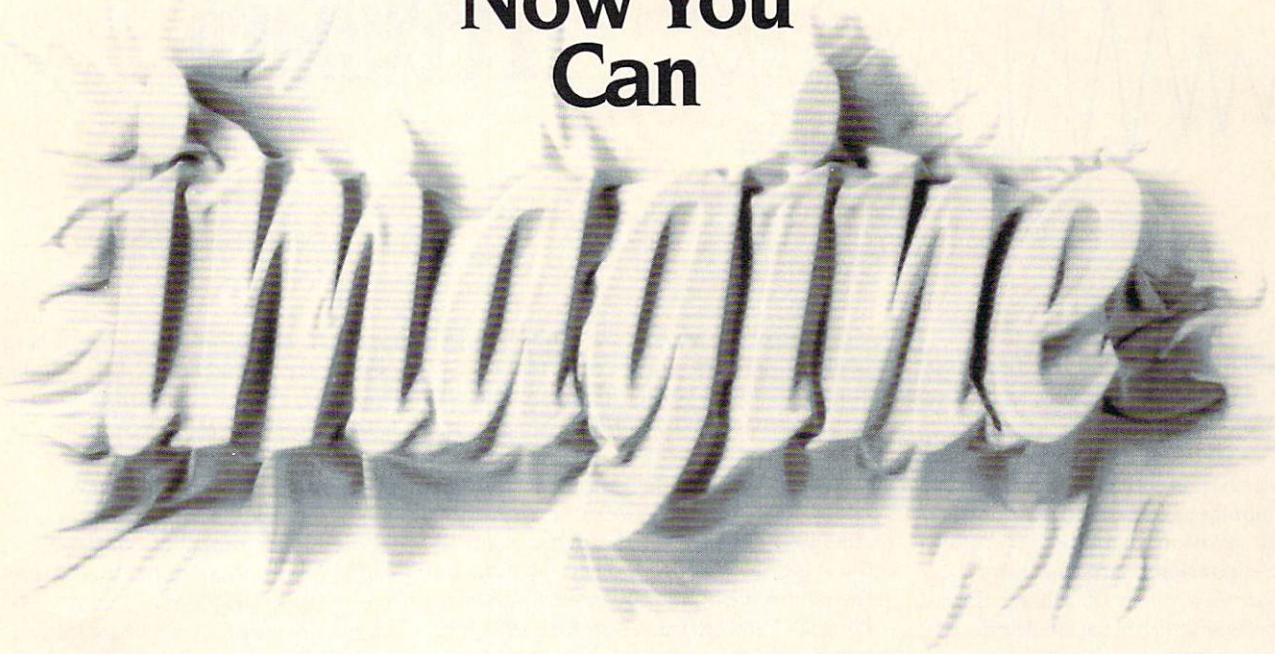
With Amiga 3000's now shipping in the United States and the promise of CDTV, Commodore is becoming the innovating computer company of the 90's. But remember, don't lose the remote control.

Sincerely,



Don Hicks
Managing Editor

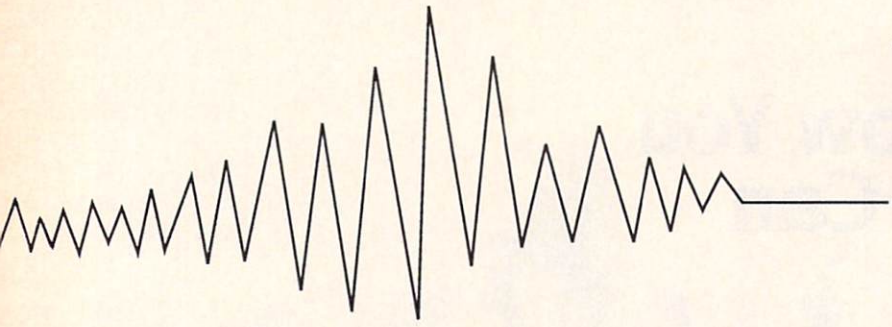
Now You Can



A Professional 3D Animation Rendering System for the Amiga

For More Information Call Impulse Inc. 612 • 566 • 0221

Circle 115 on Reader Service card.



Feedback

Dear AC:

I just finished reading the April, 1990 issue of *Amazing Computing*, and I feel compelled to write.

John Steiner writes again about problems with revision 6.1 motherboards. I read several months ago about a so-called problem with revision 6 motherboards and multiple expansion boards that turned out to be due to a batch of slightly flawed Motorola 68000 chips. That article offered a couple of solutions.

The simpler, and probably cheaper approach is to install a new CPU. I understand one costs about ten dollars. The article also stated that revision 6.2 boards are modified to accommodate any future batches of loosely made chips from Motorola. Is this the solution to some of Mr. Baebler's problems?

Oran Sands' report about possible genlock problems made me angry after I thought about it for a while. He writes that the quality of genlocked video may be affected by the fact that different members of the Amiga family put out RGB signals at different power levels. At least that is what I think he is saying. I am not an engineer or video professional. I do not own a genlock either. Certainly Commodore is responsible for the variable power levels of the RGB signals across the Amiga family, but he and the genlock makers are not totally free of blame for the problems this may cause.

As an Amiga 500 owner, I expect that if someone presents to me a product that is suitable for my machine, it will work as advertised. I expect the product to be properly designed **and tested.** I infer from Mr. Sands' article that some manufacturers never tested their products with the Amiga 500. Further, it appears that Mr. Sands either led or allowed people to believe that genlocks he was paid to review and test would work in a consistent manner on all Amigas when he did not know this to be true.

The problems this may be causing is not a result of Commodore suddenly changing the rules of the game again; I think they are the result of sloppy, unprofessional work by some manufacturers and himself. I am offended when he tells me that he and some genlock makers did not do their homework, and any problems this may cause are all Commodore's fault. **BALONEY!**

I once bought a peripheral for my Amiga 500 after I quizzed the president of the company about the suitability of his product for my hardware configuration. He assured me his product was ideal for my set-up. It did not work. I am certain the product was never even tried out with a 500. I battled this product for a couple of weeks and got it to function (like a car with a top speed of ten miles per hour). Later the company told me there was a "timing problem", as if it was an act of God. Fortunately, I think that company is out of business.

It is popular to blame Commodore for all the problems in the Amiga community. I agree Commodore deserves much of the criticism directed at it. I also know there are many dedicated and responsible Amiga developers. However, when an Amiga developer, hardware or software, does a less than thorough job of designing, making and testing his product, the developer is at fault, not Commodore. That developer must be held accountable.

Peter Margenau
Shohola, PA

—Mr. Margenau has every right to be angry about the genlock/RGB level situation. However, I think he's getting mad at the wrong people. Commodore made the mistake of assuring everyone that all Amiga models had the same signals on the RGB port. If I had a dollar for every time Commodore proudly mentioned their "NTSC standard outputs" I'd be a very rich man. The manufacturers and reviewers were unwitting dupes of these statements.

The reviewers had no reason to suspect that there was an interchange problem. The manufacturers had (in some cases) noticed the varying levels of RGB signals but no one until myself noticed the pattern across the models. I've since been thanked for these findings by several genlock manufacturers. Up till now they merely thought the varying levels were due to Amiga's lack of quality control. (This is partially true. My tests have shown that even within models the levels still vary.)

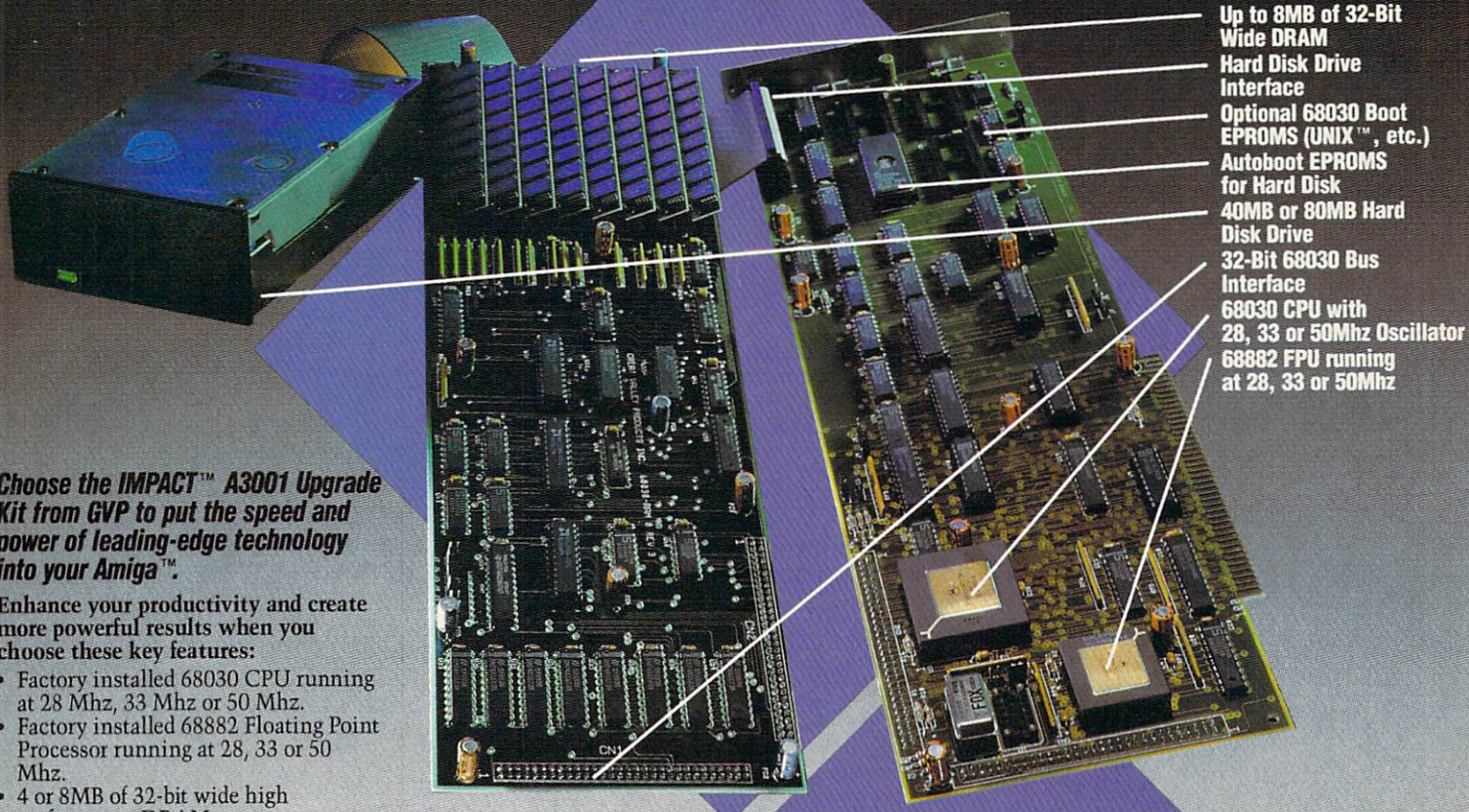
Try manufacturing a device to work properly with levels that you know are going to vary quite a bit. It isn't easy or cheap. The manufacturers tweak their products on a bench with whatever unit they use. Often a 500, sometimes a 2000. Having Commodore's assurance that the levels will be the same led them down a primrose path. Several manufacturers designed their genlocks with adjustable inputs to allow the user to cope with the problem. Some genlock's design was locked in when the only Amiga sold was the A1000 (which as shown was indeed almost to the proper levels).

Genlock quality has varied so much in the past that reviewers had no reason to suspect that there was a system interchange ➔

IMPACT A3001 UPGRADE KIT

Now Available with 50Mhz 68030 Acceleration

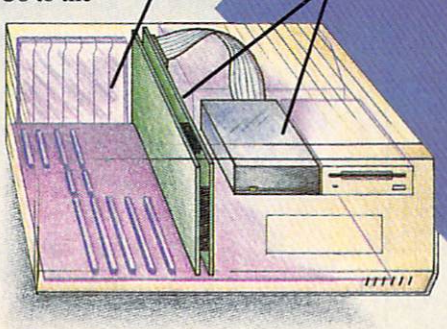
Create the fastest Amiga in the World with an A2000™ and our A3001 Kit.



Choose the IMPACT™ A3001 Upgrade Kit from GVP to put the speed and power of leading-edge technology into your Amiga™.

Enhance your productivity and create more powerful results when you choose these key features:

- Factory installed 68030 CPU running at 28 Mhz, 33 Mhz or 50 Mhz.
- Factory installed 68882 Floating Point Processor running at 28, 33 or 50 Mhz.
- 4 or 8MB of 32-bit wide high performance DRAM.
- Built-in Autobooting High Performance Hard Disk Controller with data transfer rates well over 700KB/sec.
- Quantum 40MB or 80MB hard disk drive with an average read access time of 11ms (19ms on write) and 64KB read-ahead cache. If you already own a hard disk, this item can be optional.
- Asynchronous design allowing the 68030 to run ASYNCRONOUS to the rest of the A2000 improving GENLOCK compatibility.
- ZERO SLOT SOLUTION! With the A3001 Configuration along with the bundled 40Q or 80Q Hard Disk Drive ALL A2000 EXPANSION SLOTS ARE LEFT FREE FOR FUTURE UNLIMITED EXPANSION!



When you compare, the choice becomes clear. GVP is unbeatable for price and performance.

COMPARE:	A2000 + GVP A3001	CBM A2500/30	CBM A3000™
68030 CPU	Y	Y	Y
Maximum CPU Clock Speed available & shipping *Today.*	50Mhz	25Mhz	25Mhz
Factory installed 68882 Floating Point Processor Clock Speed.	28-50Mhz	25Mhz	25Mhz
Hard Disk Controller on 68030 Processor Board.	Y	N	Y
Number of *Open* Amiga expansion slots with hard disk drive and 8MB *Fast* memory installed.	5	3	4
Allows user to start with low-cost A2000 Amiga system and grow all the way to 50Mhz 68030 performance without sacrificing anything.	Y	N	N
Brand name vendor with solid reputation.	Y	Y	Y
Typical *Ray Tracing* speed relative to a standard A2000 (28Mhz Impact)	22X	13X	13X
Fully implements 68030 Burst Mode up to 33Mhz.	Y	N	N

GVP

GREAT VALLEY PRODUCTS INC.
New Address: 600 Clark Ave., King of Prussia, PA 19406

Circle 123 on Reader Service card.

For more information, or for nearest dealer, call today. Dealer inquiries welcome.

Tel. (215) 337-8770 • FAX (215) 337-9922

problem. It would merely appear to be another genlock of "fair" quality. My tests were always done on an A1000 until recently, when I began to indicate the model of Amiga that it was designed for. Mr. Margenau should be thankful that the matter is now public.

Most genlock manufacturers are computer people making video products and they've found it isn't as easy as they once thought. MAGNI decided not to use the Amiga's RGB analog signals from the start. SuperGens were retweaked after the release of the 500 and 2000. Others may have been readjusted as well. I know of some that weren't.

Video professionals have become used to equipment that meets NTSC specs. That is considered a minimum requirement. Don't blame the messenger for the content of the message. He only delivers it.
—Oran Sands

Dear AC:

There seems to be a problem importing the .IMG clip art files into PageStream with some Hard Drive controllers. I was using a C-LTD (non-autobooting) controller without any problems. A couple of friends of mine liked the .IMG files and tried to use them. The machine would GURU every time.

Solution? A member of our user group, (A.S.L.U.G.) Amiga Support League and User Group, came by a number for the MAX TRANSFER rate in the mountlist of hard drive boot block. Replace the MAX TRANSFER number with 130560.

This has solved the GURU problem with a MicroBotics Hardframe, and a GVP Impact SCSI.

If you are having similar problems I hope that this will help.

Lloyd Campbell
Centralia, WA

Dear AC:

To all users of Deluxe Music...

First, a tip: they tell me, and I've noticed, that its printer driver is compatible only with WB1.2.

This is a call to all of us to stand up and shout to Electronic Arts to update Deluxe Music as they have DeluxePaint and DeluxeVideo. Deluxe Music is the best (and only) MIDI scoring program for the Amiga. Besides that it is terrific for composing in speed and flexibility. I've been using it daily since it came out 3 years ago. I couldn't do without it and ought to be an expert on it by now (though I only just discovered how to merge files...do a COPY (Amiga-C) on the file you want to merge into another piece of music, load that other piece, then do a PASTE (Amiga-V).

Dr. T's Copyist, the only other scoring program for Amigites, while it ends up looking more professional, is not really midi-able or very editable; you can't compose and play around in it because it's terribly tedious compared with DMCS. DMCS offers all the playfulness and options of a word processor...for music, and lets you do things you never imagined and impossible before.

So write to EA that it's a disgrace they don't/won't update Deluxe Music for the Amiga (they apparently did for Apple) to make it print proper professional-looking scores. Urge 'em to get on it, or to seek out someone who will do it for them. It doesn't need much more to be perfect; it would be worth twice the price. It's cheap now but extremely useful. The powers at EA say that there is simply no one there interested in working on it. Is there anyone else who would?

If EA can go to the moon with DPaint and DVideo let's get them to do it with DMusic. Thanks, and here's hoping.

Warner Jepson
San Francisco, CA

—A recent conversation with Electronic Arts confirmed that Deluxe Music's printer drivers are only fully compatible with WB 1.2. They are currently having trouble with WB 1.3. Also, there is no update for Deluxe Music for the Amiga planned in the near future.—Ed.

Dear AC:

Congrats to Commodore on their new A3000 computer. It's sure to find an honored place in computing. Now of course we must all get on to the business of wondering what their next one will be like! Now that we all know Commodore has those neat 2 meg Super Agnus chips, they're going to have to release a computer with its video memory running at 14.28 MHz. Of course that would enable a true HDTV output but then the Amiga's never minded being ahead of its time. Perhaps the next Amiga could even sport dual video ports to allow monitoring of the control parameters of the video program it creates. Anyone for stereo video binoculars, 2 screen games or 2 person computing at home? And does anyone wonder why the system expansion bus ends at the cover of the 3000? Maybe the truth is that the 2000 series is limited to only 7 cards!

C. Robert Spencer
Spencerport, NY

Dear AC:

I am preparing to go to the U.K. and have a question concerning my Amiga 500. When I arrive, will I be able to simply buy a British Amiga power supply and start up my computer, or will I have to use a step-down transformer?

Bobby R. Edmonson
Honolulu, HI

—When you reach your destination in the U.K. contact one of the local Amiga dealers and ask them if you need a power converter or a new power supply. Or, contact Commodore International in the U.K. (011-44-81528-9869). They will lead you in the right direction.—Ed.

•AC•

All letters are subject to editing. Questions or comments should be sent to:

Amazing Computing
P.O. Box 869
Fall River, MA 02722-0869
Attn: FEEDBACK

Readers whose letters are published will receive five public domain disks FREE.

Apples, Oranges, Accelerators, & MIPS

68030 Accelerators for the Amiga 2000

by Ernest P. Viveiros, Jr.

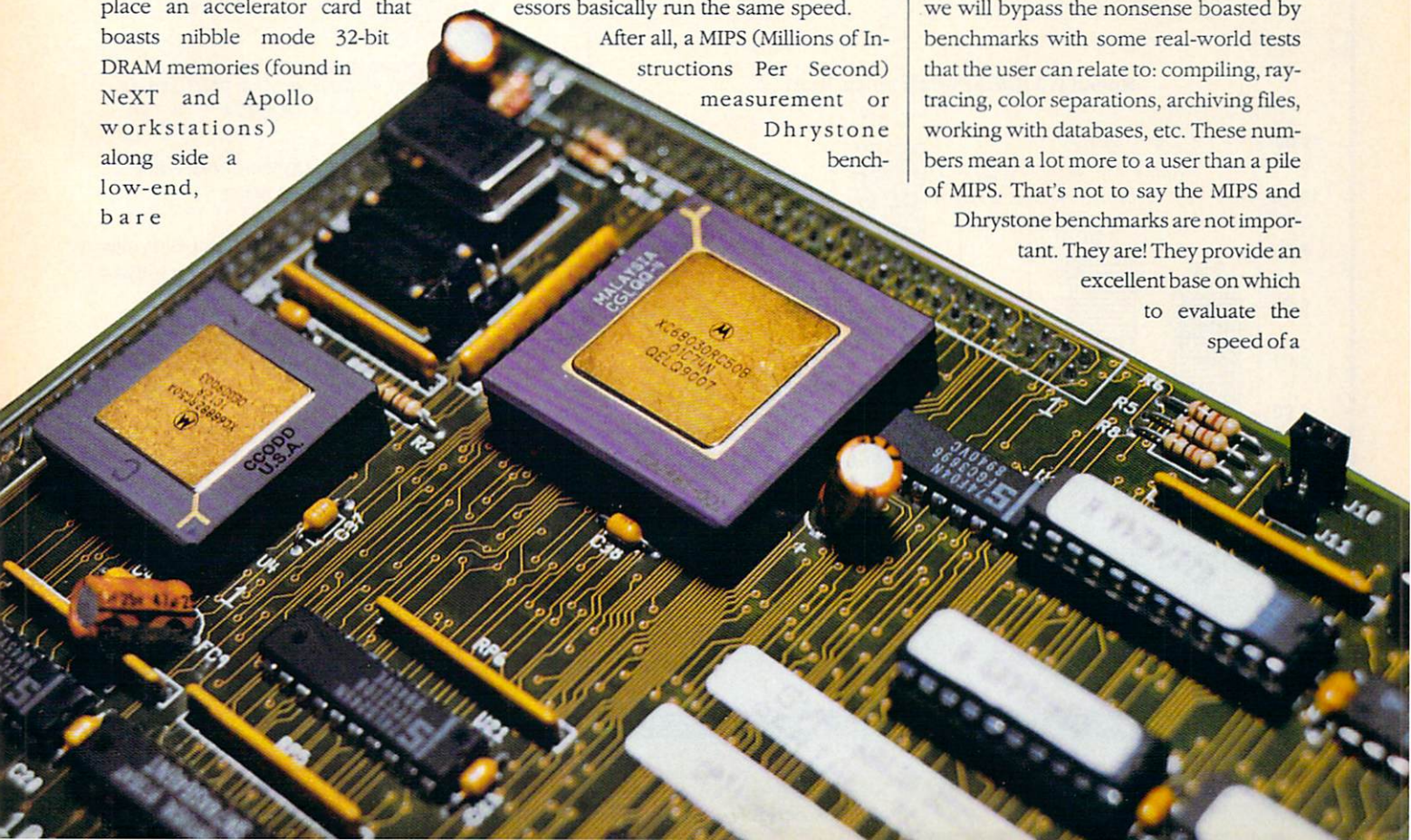
MOST AMIGA USERS SEE THE new A3000 and wonder why they will ever need an A2000. When discussing speed alone, it would appear that the new A3000 would easily pass the A2000. Yet, existing A2000's may be modified by the many accelerator cards available. So a simple test of a stocked A3000 seemed absolutely necessary.

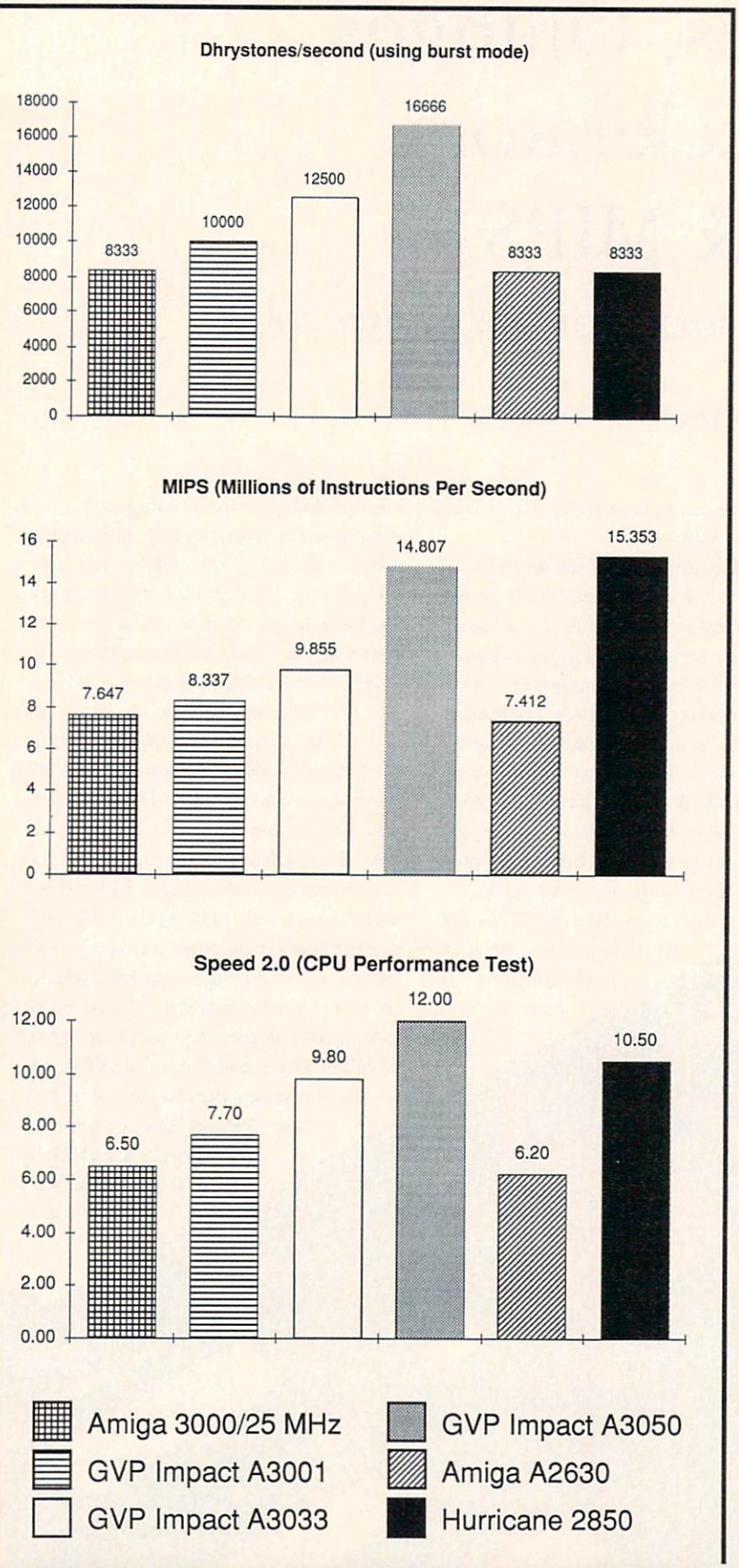
However, trying to compare all of the 68030 accelerators for the Amiga 2000 would be like comparing apples to oranges to bananas. It would be unfair to place an accelerator card that boasts nibble mode 32-bit DRAM memories (found in NeXT and Apollo workstations) along side a low-end, bare bones processor accelerator that is just a chip and little else.

This article is not meant to be the complete guide to accelerators. Instead, it is intended to open up to question and review the various options and configurations available in the Amiga 68030 accelerator market. You will see that the bottom line shouldn't be which accelerator is the fastest, at which price. After all, any 33 MHz 68030 accelerator is going to be considerably faster than the same chip running at 28 MHz, and that all like processors basically run the same speed.

After all, a MIPS (Millions of Instructions Per Second) measurement or Dhrystone benchmark probably means nothing to a novice programmer who needs processing speed and hard disk access to speed up the software development process; or an artist who needs raw processing power to speed up the creation of animations, or even Joe Amiga who wants to add a little pep to his everyday efforts.

Each individual has different needs (including budget constraints) which would be satisfied by specific processors with specific configurations. It is our goal that after individual accelerator reports we will bypass the nonsense boasted by benchmarks with some real-world tests that the user can relate to: compiling, ray-tracing, color separations, archiving files, working with databases, etc. These numbers mean a lot more to a user than a pile of MIPS. That's not to say the MIPS and Dhrystone benchmarks are not important. They are! They provide an excellent base on which to evaluate the speed of a





main processor. In fact we have included a chart of the MIPS and Dhrystone benchmarks (as well as one of Speed 2.0) for the accelerators we examined. [See charts at left.]

Let's get down to brass tacks. We are going to look at accelerators from three of the companies which produce 68030 accelerators for the Amiga 2000: Great Valley Products, Imtronics, and Commodore-Amiga.

First let's take a look at the Impact series of accelerators from Great Valley Products (GVP). Their accelerator products for the Amiga 2000 include:

GVP A3001-4MB/0
28 MHz 68030/68882, 4 MB 32-bit RAM, without hard drive. $2299.00

GVP A3001-4MB/40Q
28 MHz 68030/68882, 4 MB 32-bit RAM, with 40MB Quantum AT hard drive. $2799.00

GVP A3001-4MB/80Q
28 MHz 68030/68882, 4 MB 32-bit RAM, with 80MB Quantum AT hard drive. $3199.00

GVP A3033-4MB/0
33 MHz 68030/68882, 4 MB 32-bit RAM, without hard drive. $3199.00

GVP A3033-4MB/40Q
33 MHz 68030/68882, 4 MB 32-bit RAM, with 40MB Quantum AT hard drive. $3699.00

GVP A3033-4MB/80Q
33 MHz 68030/68882, 4 MB 32-bit RAM, with 80MB Quantum AT hard drive. $3999.00

GVP A3050-4MB/0
50 MHz 68030/68882, 4 MB 32-bit RAM, without hard drive. $4399.00

GVP A3050-4MB/40Q
50 MHz 68030/68882, 4 MB 32-bit RAM, with 40MB Quantum AT hard drive. $4899.00

GVP A3050-4MB/80Q
50 MHz 68030/68882, 4 MB 32-bit RAM, with 80MB Quantum AT hard drive. $5199.00

The Impact accelerators from GVP offer the following as standard options along with the factory installed 68030/68882 processors:

- 4 MB of 32-bit nibble mode DRAM—supports up to 8 MB on board. (which allows full support of the 68030 burst mode).
- Built-in autobooting hard disk controller (with data transfer rates over 700KB/sec.).
- Asynchronous design to improve genlock compatibility.
- Zero slot solution (the processor, the DRAM, and the hard disk controller use only one slot—the coprocessor slot).

Options include your choice of 28MHz, 33MHz, and 50MHz processors, as well as a 40MB/80MB (11 ms w/64K read-ahead cache) Quantum hard drive.

➔

OVER 100,000 Satisfied Customers

NOW EVEN LOWER PRICES!!!

MEMORY

M501 A500	79.95
Starboard 1MB	289.95
Starboard 2MB	399.95
8-UP with 2MB	289.95
8-UP with 4MB	449.95
8-UP with 6MB	599.95
8-UP with 8MB	749.95

COMMODORE AMIGA

System Packages

We customize *AMIGA* 2000 and *AMIGA* 3000 desktop video systems to meet your individual needs. Call and talk to one of our system specialists to get the best price for your requirements.

GVP
GREAT VALLEY PRODUCTS INC.

28 MHz 68030 28 MHz 68882	33 MHz 68030 33 MHz 68882
with 4 MB of 32 bit memory **1949.95**	with 4 MB of 32 bit memory **2549.95**
with 40 MB hard drive **2399.95**	with 40 MB hard drive **2899.95**
with 80 MB hard drive **2699.95**	with 80 MB hard drive **3199.95**

HARD DRIVES

GVP Hardcard /0	159.95
GVP A2000-2/0	179.95
GVP A2000-8/0	199.95
HardFrame	219.95
Hardcard 40MB	**499.95**
Hardcard 80MB	699.95
Hardcard 105MB	799.95
Quantum Pro 40	**379.95**
Quantum Pro 80	649.95
Qunatum Pro 105	699.95
Supra 105/1000	1099.95
Supra 105/500	999.95
Supra 40/1000	683.95
Supra 40/500	599.95
Supra 80/1000	949.95
Supra 80/500	875.95
Supra Wordsync	169.95

VIDEO

Flicker Fixer	469.95
Framebuffer with 1MB	749.95
Magni 4004S with remote	1629.95
MiniGen	189.95
Panasonic 1410	209.95
Panasonic 1500	299.95
Panasonic Vari-Lens	39.95
Polaroid Freezeframe	1599.95
ProGen	349.95
Sharp JX-100	749.95
SuperGen	669.95
SuperGen 2000	1699.95

SOFTWARE

Deluxe Paint III	89.95
Digi Paint	64.95
Digi View 4.0	129.95
Digi Works 3D	79.95
Diskmanager MAC	79.95
Homebuilder's CAD	119.95
Lattice ++	249.95
Pagestream 2.0	179.95
Prowrite 3.0	99.95
Saxon Publisher	249.95
Soundscape Pro MIDI	109.95
Turbo Silver	109.95

CALL FOR UNLISTED TITLES

FDATA-10 99.95 | **FDATA-20** 239.95 | **HP PAINTJET** 1099.95

Government and School Purchase Orders Accepted.

Circle 114 on Reader Service card.

InterComputing, Inc.

2100 N Hwy 360, Suite 2101, Dallas, TX 75050-1015

1-800-622-9177

Customer Service: 214-988-3500

InterComputing Deutschland Inc.
Schönbecker Str. 55-57 Telefon: 0202/89155
5600 Wuppertal-2 Telefon: 0202/89304

InterComputing France
34, Avenue des Champs Elysees Phone: (1) 42821603
75008 Paris FAX: (1) 42806649

As always we have the most 'customer friendly' terms: S/H $4.95 in cont. USA; $30.00 min. order ; MASTERCARD & VISA with NO credit card fee; in Texas add 7% Sales Tax. $12.00 shipping to APO/FPO addresses. RMA# required on all returns. All prices subject to change.

MasterCard VISA

A Look at CSA's Mega-Midget Racer

AT PRESS TIME, WE RECEIVED AN interesting accelerator product from Computer Systems Associates, Inc., (CSA)—the Mega-Midget Racer. The first thing noticeable about this accelerator is that it doesn't plug into the coprocessor slot in an A2000. Instead, the Mega-Midget Racer plugs into the 68000 processor socket in an A2000 or A500. The replaced 68000 is resocketed on the Mega-Midget Racer, while the processor is software-selectable. 100% compatibility is boasted by using the on-board 68000.

You could call the Mega-Midget Racer the 'What You Want is What You Get' accelerator. The basic Mega-Midget Racer comes without a processor. You can choose from options which include a 68030 running at 20, 25, or 33 MHz. Also optional is a 68881/68882 math coprocessor, which can be clocked by either the processor's CPU or even a second faster one (up to 50MHz). Finally, for you speed demons, an optional high-speed 32-bit, 512K SRAM bank is available. The Amiga ROM Kernel is copied into and executed out of this high-speed, 32-bit SRAM. This is sure to liven up a sluggish Amiga.

It is often said that you get what you pay for. With the Mega-Midget Racer's low price, that statement goes right out the window! The engineering put into this product is commendable, including the notable use of surface-mount technology to enhance reliability. This preliminary examination of the Mega-Midget Racer proves it to be an interesting entry into the A2000 accelerator market. It offers all of the *basic* features that users want of an accelerator, at an unheard of low price. You'll be hearing more about the Mega-Midget Racer, both from *Amazing Computing*, and the many satisfied customers. *—EPVjr*

Mega-Midget Racer without Processor: $675.00
Mega-Midget Racer with 20-MHz 68030: $795.00
Mega-Midget Racer with 25-MHz 68030: $895.00
Mega-Midget Racer with 33-MHz 68030: $1095.00

Computer Systems Associates, Inc. (CSA)
7564 Trade Street
San Diego, CA 92121
Phone: (619) 566-3911
FAX (619) 566-0581
Inquiry # 334

Next, let's take a look at the Imtronics' Hurricane series of accelerators for the Amiga 2000:

Hurricane 2800-0KB
28 MHz 68030/68882, 0KB 32-bit DRAM.
$1195.00

Hurricane 2800-4MB
28 MHz 68030/68882, 4MB 32-bit DRAM.
$1995.00

Hurricane 2800-4MB/40Q
28 MHz 68030/68882, 4MB 32-bit DRAM, 40MB Quantum SCSI hard drive. $2495.00

Hurricane 2800-4MB/80Q
28 MHz 68030/68882, 4MB 32-bit DRAM, 80MB Quantum SCSI hard drive. $2595.00

Hurricane 2850-0KB
50 MHz 68030/68882, 0KB 32-bit DRAM.
$3295.00

Hurricane 2850-4MB
50 MHz 68030/68882, 4MB 32-bit DRAM.
$3995.00

Hurricane 2850-4MB/40Q
50 MHz 68030/68882, 4MB 32-bit DRAM, 40MB Quantum SCSI hard drive. $4495.00

Hurricane 2850-4MB/80Q
50 MHz 68030/68882, 4MB 32-bit DRAM, 80MB Quantum SCSI hard drive. $4595.00

The Hurricane accelerators from Imtronics offer the following as standard options along with the factory-installed 68030/68882 processors:

- 1 to 16MB 32-bit DRAM (Upgradable in 1MB increments if using 256K x 4 chips or 4MB increments using 1M x 4 chips).
- 'Fast' 750KB/sec SCSI hard disk controller on-board. Optional upgrade to the new and faster SCSI II standard is available.
- Asynchronous design to improve genlock compatibility.
- Zero slot solution (the processor, the DRAM, and the hard disk controller use only one slot—the coprocessor slot).

Finally let's take a look at the A2630 board from Commodore-Amiga.

A2630
25 MHz 68030/68882, 2 MB 32-bit RAM.
$2195.00

The only feature the A2630 offers as standard options along with factory-installed 68030/68882 processors is the 2MB 32-bit memory. This is expandable to 4MB using standard ZIP type DRAMS.

INSTALLATION & DOCUMENTATION

In our preliminary testing, we have found all accelerators to work effectively—once they were installed. The technicians who test and write accelerator articles all too often forget that the average power user (for example, an artist who needs ray-tracing power) is not all that technically motivated. He reads an article which says that the installation is a simple matter of inserting a card. He can handle that, so he pulls out his trusty credit card and orders. When the package arrives he zealously rips apart the packaging, eager to use his newly bought power. However, he quickly finds himself with a manual filled with technical terms, an accelerator card with EPROMS that have to be installed, not to mention the endless jumpers that have to be set. And then there's the software. All this can be very scary for a non-technical person, and will often result in costly calls to the accelerator company's technical support line.

This is not always the case, but too often it is. We often forget how important good documentation, and tech support is, especially with products that you are paying thousands of dollars for. Then there is the fact that some of the configurations that you can buy don't come prepared or preassembled. You could probably put it together with a good manual, but too often the manual is insufficient for an average user. Once again, this might not bother the technophile, but it would frighten the average user (although, sometimes even technophiles are frightened at these packages and manuals).

The point is not that the product is bad because you may have a hard time assembling it, but maybe you should purchase it from a dealer and have him install it (he'll also be there to support you if you run into any compatibility problems later).

Finally, there is the software and hardware compatibility problem. For the most part, most software will run with the Amiga in the 68030 mode. The majority that won't will run in the 68000 mode. However, there is some software and some interface hardware that won't run with some accelerators. It's not very satisfying to buy an accelerator and then find that it won't work with your favorite program. You would be well advised to contact the manufacturer about any software & hardware incompatibilities.

With all this in mind, let's take a closer look at these accelerator packages:

THE GVP IMPACT ACCELERATORS

We were pleasantly surprised upon opening the boxes for the GVP Impact boards. The GVP Impact accelerators without hard drives come preassembled, and pre-jumpered. The installation is flawless: just plug the board into the coprocessor slot and turn the machine on—that is, in most cases. In rare cases, the board is improperly jumpered, or a jumper might come loose in transit. However, a quick call to the helpful people at GVP tech support line can easily resolve this minor problem.

*(*Note! If your A2000 motherboard is one of the first German-made motherboards (4 layer), you are going to have to remove your 68000 from the motherboard, regardless of which board you use. This is not a simple task, considering that the 68000 is situated under the power supply/disk drive. Most Amigas shouldn't need this, but be forewarned.)*

(continued on page 95)

NEW PRODUCTS & other neat stuff

compiled by E.G. Fedorzyn

PUTTING THINGS IN PERSPECTIVE

For those looking to back up their words with something of substance, **Mindware International** might be able to offer some assistance. The company's new program, ***3D Text Animator***, allows for the creation of 3D text animations in the ANIM format.

In this veritable 3D text "construction kit", users can, through a simple point-and-click interface, enter a text string, select a font, resolution, colors, light direction and intensity, and animation pattern. Frames for the text animation will then be generated according to the pattern chosen. Users can mix and match patterns for 3D animation in three categories: text animating into the display, animating out of the display, and animating at the center of the display. Resulting ANIMs can be easily appended using the program's ANIM Cut-and Paste features.

With *3D Text Animator*, 3D fonts from other programs such as *Sculpt*, *Turbo*, or *Videoscape* may be imported. The program also allows any Amiga 2D bitmapped font to be converted to 3D, enabling it to then be used as a font for 3D text animation. The provided 3D Font Editor enables these converted fonts to be "smoothed" prior to usage.

Results achieved may be genlocked over a video source, or overlaid on a background using external tools.

3D Text Animator, *1MB of RAM required, Price: $49.95, Mindware International, 110 Dunlop W., P.O. Box 22158, Barrie, Ontario, Canada L4M 5R3, (705) 737-5998. Inquiry #320*

THE MOVERS & THE FLICKERS

MicroWay has announced the release of ***DEB 2000***, their Denise Extender Board for the A2000 and 2500. The board allows MicroWay's flickerFixer, the graphic enhancer for the A2000 and 2500, to be run in the Amiga without utilizing the video slot, thereby leaving the slot free for other devices such as internal genlocks and frame buffers.

DEB 2000 transfers the video signals required by the flickerFixer from the Denise socket on the Amiga motherboard to the DEB connector board. The flickerFIxer is then positioned behind the existing XT slots and connected to the DEB 2000 via a cable.

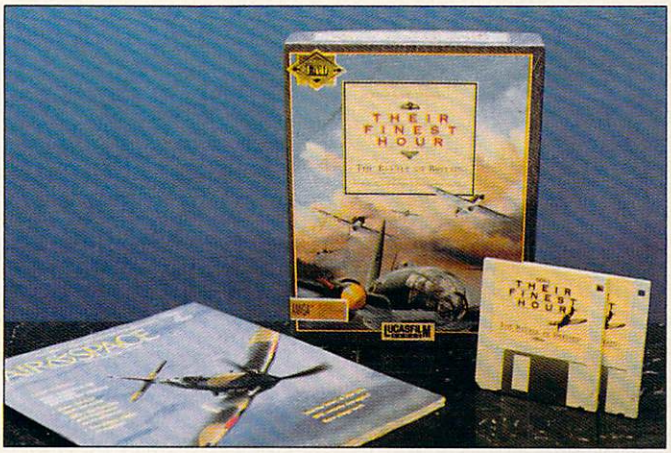
Also, MicroWay has reduced the retail price of the *flickerFixer* from $595.00 to $495.00. *flickerFixer* is compatible with AmigaDOS 2.0 and the Enhanced Chip Set from Commodore.

DEB 2000, *Price: $75.00, MicroWay, P.O. Box 79, Kingston, MA, 02364, (508) 746-7341. Inquiry #322*

MISSION: ACCOMPLISHED

LucasFilms has released the Amiga version of their newest flight simulator/strategy game, ***Their Finest Hour***. Based on the exploits of the early days of WWII, this game provides both the Luftwaffe and the RAF perspective to some of the most daring dogfights of any war. With over 52 possible missions and a contingent of eight different aircraft, Amiga flight jockeys can now either defend the mighty shores of Great Britain or prepare the isles for the German invasion campaign, Operation Sea Lion.

From the same people who developed *BattleHawks 1942*, *Their Finest Hour* combines cockpit realism with a variety of missions and antagonists to produce a game both entertaining and informative. Operation aircraft can be British fighters, German fighters, or German bomber planes—each plane has its own independent characteristics and requires a different style of combat. Each level of combat is more difficult, and repeated successful completions of missions are rewarded with medals or promotions. Characters can be developed to advance through the ranks. As a final touch, LucasFilms has once again supplied an excellent manual which delves into the history and the people of the time, and provides enough in the way of detailed aerial techniques to inspire any good Amiga air warrior.



Their Finest Hour, *Price: $59.95, produced by LucasFilms Games, distributed by Electronic Arts, 1820 Gateway Drive, San Mateo, CA 94404, (800) 245-4525. Inquiry #324*

→

PICTURE THIS

From the folks who brought you *Doug's Math Aquarium* now comes ***MathVision***, a new math and scientific visualization program.

Seven Seas Software's latest release features data input/graphics output, HAM, overscan, half-brite, ARexx support, image processing, IEEE library with math coprocessor support, and hooks to access external programs. According to Seven Seas President Otto Smith, the program "offers an entire panorama of gadgets, controls and functions for visualization. The power of the program has far exceeded [Seven Seas'] original expectations."

Registered users of *Doug's Math Aquarium* may purchase *MathVision* at a special upgrade price.

MathVision, *Price: $197.00, Upgrade price: $30.00, Seven Seas Software, P.O. Box 1451, Port Townsend, WA 98368, (206) 385-1956. Inquiry #323*

DIAMONDS ARE FOREVER

Time to don the ol' cleats and slap a big wad o' bubble gum in your mouth (just a pinch between your cheek and gum), and get set for some hardball, Accolade-style.

New from **Accolade** is ***Hardball II***, a sequel to *Hardball!*, their popular baseball simulation originally released in 1985. The company has reportedly acted on users' comments and requests stemming from the original game to produce this new and improved baseball simulation.

In *Hardball II*, you assume the role of team manager, supervising the players and controlling their on-screen performance, including batting order, lineups, substitutions, and position swaps. You can select teams from the preprogrammed library available, or use the Team Editor to either create new players with their own stats or to enter stats from real-life professional players. The Team Editor also lets you mix and match players and teams. Statistics are automatically updated at the end of each play to give you the same cutting edge as the Lasordas and Morgans of the baseball world.

But before you start kicking dirt in the umpire's face, you might want to first review any controversial calls with *Hardball II*'s Instant Replay option. Other enhanced features include the choice of six simulated major league stadiums including Boston, Kansas City, and Toronto, as well as the choice of five different field views including upper deck overview, behind the plate, and right and left infield. The game's complexity is set by sixteen different options, so anyone from Little Leaguer to veteran can take a swing.

Hardball II, *Price: $49.95, Accolade, 550 South Winchester Boulevard, Suite 200, San Jose, CA 95128, (408) 985-1700. Inquiry #321*

1. Drink from the Fountain of Youth.

2. Receive a perfect 4-star rating on "Star Search".

3. Read every issue of *Amazing Computing* ever published.

We've just made your life's goals one easier.

No doubt you have over the years compiled a list of goals you hope to achieve. And certainly *Amazing Computing* holds a prominent position on that list. That's why we've made it that much easier for you to acquire the complete *Amazing* library at terrific savings. For a limited time, you may purchase volume sets of AC at

a savings of over 50%!

AC Volume 1 is now available for just

$19.95*!

(A regular $45.00 value, this first year of AC includes 9 info-packed issues.)

AC Volumes 2, 3, & 4 are now yours for just

$29.95* each!

(Volumes 2, 3, & 4 include 12 issues each and regularly sell for $60.00 per volume set.)

PLUS! We're now offering subscribers freely redistributable disks** at distribution prices. Now's the time to complete your Fred Fish, Amicus, or AC disk collection. Pricing for subscribers is as follows:

1 to 9 disks: $6.00 each
10 to 49 disks: $5.00 each
50 to 99 disks: $4.00 each
100 disks or more: $3.00 each

(Disks are $7.00 each for non-subscribers.)

To order volume sets, freely redistributable disks, as well as single issues, use your Visa or MasterCard and

call 1-800-345-3360.

Or just fill out the order form insert in this issue.

Amazing Computing and freely redistributable software—at savings beyond your wildest dreams.

*Postage & handling for each volume is $4.00 in the US, $7.50 for surface in Canada and Mexico, and $10.00 for all other foreign surface.
**AC warranties all disks for 90 days. No additional charge for postage and handling on disk orders. AC issues Mr. Fred Fish a royalty on all disk sales to encourage the leading Amiga program anthologist to continue his outstanding work.

GETTING HITCHED

Next month, **Interworks** will begin offering the ***ENLAN Network System***, a DECnet networking system for the Amiga. A compatible implementation of DECnet protocols (defined by the Digital Equipment Corporation), the system provides the ability to exchange files and data with DEC VAX/VMS computers, other Amigas, and machines running the DECnet protocol.

Among its features, the Interworks system provides a File Copy Utility, File Directory Utility, File Delete System, and File Access Listener program. The system's Network Virtual Terminal facility allows multiple, simultaneaous log-in sessions to one or more remote VAX/VMS systems. With this facility, most terminal emulator programs can be used with ENLAN. (Interworks supplies a VT100 terminal emulator program with the package.)

The ENLAN Network System supports both thick and thin-wire Ethernet on the A2000 and higher machines, and asynchronous DECnet on all Amiga models using the serial port. The system also has the capacity for dial-out and dial-in operation via modem.

Versions of the ENLAN Network System start at $295.00

Interworks, 195 E. Main Street, Suite 230, Milford, MA 01757, (508) 476-3893. Inquiry #325

AN INTEGRAL PART

Integral Systems Co. has announced the release of a new series of Amiga products that will support the video professional. ***VidControl*** and ***MasterControl*** are the first two programs to be released as part of the Ohio-based company's video production integration package.

VidControl provides the capability to control any Amiga application via a signal applied through GamePort #1. Basically, any application that normally accepts keyboard input may be managed using the *VidControl* program. Among its features are support for user-definable response to input signal, as well as variable time delays between the input event and the passing of the message to the application.

MasterControl provides all the capabilities of *VidControl*, but will accept six discrete inputs from the gameport, as opposed to the one used by *VidControl*.

VidControl: $30.00; MasterControl: $50.00. *Integral Systems, P.O. Box 31626, Dayton, OH 45431, (513) 237-8290. Inquiry #335*

CORRECTION!

Last month in "New Products" (V5.6), it was reported that MicroTouch Systems' *Amiga TouchDriver* was the first touch screen to provide a two-button mouse emulation touch screen for the Amiga. It has come to our attention that **Future Touch Inc.**, a joint venture of Amigo Business Computers and Business Technology Services, has been marketing a two-button mouse emulation touch screen for the Amiga since June of 1987.

Our apologies to Future Touch, Inc. and to our readers for this error in reporting. For more information on Future Touch systems, contact *Future Touch, Inc., 192 Laurel Road, East Northport, NY 11731, (516) 757-7334. Inquiry #326*

Other Products Received

Budokan
Price: $49.95
Electronic Arts
1820 Gateway Drive
San Mateo, CA 94404
(800) 245-4525
Inquiry #327

European Challenge
Price: $21.95
Accolade
50 South Winchester Boulevard
Suite 200
San Jose, CA 95128
(408) 985-1700
Inquiry #328

Heat Wave
Price: $44.95
Accolade
50 South Winchester Boulevard, Suite 200
San Jose, CA 95128
(408) 985-1700
Inquiry #329

Stryx
Price: $34.95
Pygnosis
122 Century Buildings, Tower Street
Brunswick Business Park
Liverpool, England L3 4BJ
(051) 709-5755
Inquiry #330

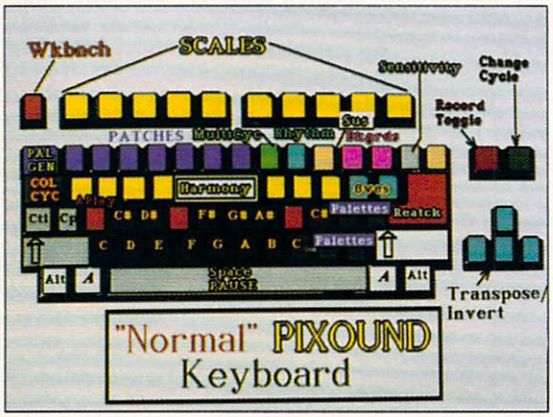
Senses Working Overtime:

PIXOUND 2.1

by R. Shamms Mortier

I SUFFER UNDER THE VELVET whip of two muses, music and visual art. With that conscious bondage in mind I purchased an Amiga, in the hopes that both addictions could be at least addressed if not totally satisfied. I have never been sorry, except when I realize that I get about half of the sleep that non-Amigans do. There has been a missing ingredient though, and it's one that I have even fantasized writing a program for when I retired from present tasks. It is the ability of the Amiga to actually "play" visual screens, interpreting the visual information into music by filtering it through some exclusive algorithm. But alas, someone has beat me to the punch! That "someone" refers the creative folks at Hologramophone, so that at a cost of about $100.00 (U.S.), you can now "hear" what your eyes are seeing with PIXound. The Amiga can now "play" visual screens, and direct the output to MIDI synths. For decades, experimental composers have been "writing" music by using color and graphic symbols, and asking musicians to interpret the results. This software makes you the composer, and

Straight key assignments of the PIXound keymap

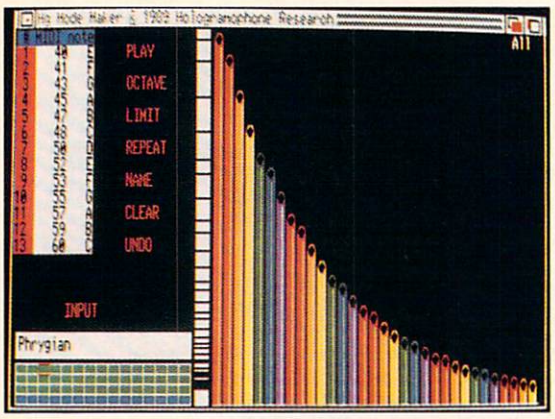


Serious Jammin':

HYPERCHORD

by Howard Bassen

ALL MUSICIANS OCCASIONALLY get stuck in creative ruts, limited by their habits, skills and style. At times like these, they tend to fall back on their favorite riffs subconsciously when jamming or composing. Hyperchord—a new algorithmic composer/dynamic riff sequencer from Hologramophone Research—allows you to stretch the limits of your own musical ability and break new musical ground, through the creation, storage and manipulation of riffs in real time.



Hyperchord's Mode Maker

THE LOOK AND FEEL

All of the design screens and menus are rendered in a very low-resolution graphics mode, probably to save chip memory space. The vertical axis represents the musical scale, or pitch, while the horizontal axis represents time. Notes can be "drawn" or entered on the grid manually, or they can be generated automatically from two menu-selectable "grab bags" each putting 30 short, fairly unconventional sequences or riffs in selectable memory locations or "riff banks". Unlike most Amiga music software, there are no "demo" songs (other than the grab bags that can be run to show off the full capabilities of the program).

CREATING AND EDITING RIFFS

In the Draw mode, notes are entered "freehand", either as a line between two points, or on the grid with the mouse. As you move the mouse up and down, letters flash along the left side of the screen, indicating the notes located along each vertical axis of the grid. You hear the pitch of each note as you move the mouse. A feature lacking here is a piano keyboard gadget that displays the note(s) being selected. The text message "A#" accompanied by the sound of the note just doesn't register in my mind as well as the sight of a key being selected on a piano keyboard might.

In experimenting with Hyperchord, I entered patterns in the Draw mode, and set

the Amiga/MIDI connection the orchestra. You can either ask the computer to play a graphic screen, or you can involve your own sensibilities to intervene in the making of the music (by taking control of the movements of the mouse).

If you desire, you can create your own graphics screens in 32 color lo-res, translate them into a format that PIXound can "read" (with an on-board module), and then sit back and watch/listen to the results. There is a long list of graphics screens already configured and stored on the disk for instant gratification. The visual designs are based upon both the hue and intensity of color, because it is these parameters that PIXound uses to translate visual information into sound. The principle is simple, but the results are truly amazing. To add to your musical joy, the whole keyboard is mapped out with different ways that you can interact (in real time) with the music. My absolute favorite is that the function keys are dedicated to certain scalar patterns (augmented scales, Major/Minor scales, modes, and other patterns) that automatically remap the visual so that it responds to color and outputs dedicated scale patterns. The interaction is limitless, and will definitely promote an inter-disciplinary attitude in the arts. Musicians will be investigating color and visual form, and visual artists will be experimenting with the amplitude and waveform possibilities of their paintings. When you hear the partially controlled partially random results through a good MIDI device, you will surrender the few hours of sleep that the Amiga has left you with. And if, like me, you are both a visual and musical addict, PIXound will have you walking in the Seventh Heaven.

HOW IT DO THE MAGIC IT DO

To begin, PIXound advertises itself as a "MIDI musical art interpreter". I use it in conjunction with the "Midia Musicbox", and also with my Casio-1000 synthesizer. You can have it playback Amiga sounds alone however, if you don't have a MIDI device. But since there's no way to load in your own Amiga samples, you'll have to settle for the rather bland samples on board.

There's a whole host of new Amiga videographic ware that addresses creativity in a fashion similar to PIXound. All of it except PIXound, however, works with graphics, not sound. The basic process is to use the Amiga keyboard as a macro converter, so that by depressing a key or a combination of keys, you can cause a defined action to take place. With Elan's "Performer", this defined action is the appearance of a still or an animation that you have loaded previously. With PIXound, it is a manipulation of the Amiga soundchip or a MIDI command sent to a synth or soundbox. PIXound 1.0 was a rather enjoyable but simplistic affair. It was easy to "play" the picture before you because the commands were limited. After a cursory reading of the manual, you were anxiety free and on your own. PIXound 2.1 has a myriad of additional options, and the "cost" is a longer study of the manual in order to become even quasi-familiar with expected results. PIXound 1.0 was a nice toy. PIXound 2.1 has professional and performance applications.

There are really three keyboard maps that have to be referenced, as I have attempted to show in my illustrations: The first is a general mapping of the standard keyboard keys, the second shows applications addressed by Shift/Key functions, and the third is a roadmap of more complex Alternate and Amiga-key combinations. There are far too many options to rely on extemporaneous playing around. Either you will work with the manual in front of

---

the loop mode gadget to "on" to hear the sequences (riffs) thus created. I found that when a single note on the grid is changed the music stops, necessitating a restart for the modified riff to continue. This makes interactive editing harder than in the uninterrupted, continuous looping mode used in "M" and some other sequencers. Continuous looping lets the user hear the effect of changing a single note immediately.

Editing notes in Hyperchord's Design Screen grid requires concentrated effort and precise aim. Clicking anywhere but exactly in the center of the tiny squares given may change or erase notes in the column to the left of those truly wanted.

In contrast to the manual editing mode with its weaknesses mentioned above, Hyperchord's automatic pattern-generating and modifying features are the most full-featured of any music program that I have seen. There are gadgets on the screen that let you smear, rotate, randomize, mix, and otherwise process the entire pattern on the design screen, or any selected set of neighboring notes.

But before "massaging" a newly-created riff, it is important to hear exactly how it sounds. The length can be changed by dragging a section selector to choose any number of adjacent notes, so only those notes of the riff will play when the spacebar is depressed to begin. Once a pattern or riff is finished, it can be saved in the "riff bank" as one of 30 preset riffs that are accessible from the Amiga keyboard. Store riffs by simply clicking the mouse-pointer on a separate letter of the replica of the Amiga keyboard displayed at the bottom left of the design screen.

PREVIEWING RIFFS

To preview the riffs that I created, I used the internal sound capabilities of the Amiga. The only internal voice that appears to be available on the single disk that comes with the software package is a default sound like that of a piano.

However, several other default sounds are provided as simple waveforms. These are available from a menu-selectable window called "Voice Group". This window provides a choice of sine, square and triangular waves for the internal Amiga voices. No explanation of this feature is provided in the sparse documentation that comes with the software.

Also, unlike most music programs, the "instruments" drawer is empty. The riffs I previewed tended to sound somewhat flat, due to the single piano timbre. The other internal voices (sine, square, and triangular) made my riffs sound rather abstract, like "electronic music" beeps produced by earlier home computers. I wondered whether a good assortment of internal Amiga sounds (instruments) would improve the quality of the riffs.

I tried several Amiga IFF Instruments from other programs (replacing the piano timbre as voice 10 in the voice group window). The manual shows the release (decay) times of the internal Amiga instruments to be controllable by the backslash key on the Amiga keyboard. However, I could not change this parameter noticeably with any Hyperchord controls, so I was not able to give the internal voices an echo-like, sustained quality with long decay times. This and other types of "envelope control" are nearly-standard features found in most Amiga music programs.

You can attempt to spice up your riffs by using any of the 52 rhythms selectable at

you, or you will be forced to do a bit of planning and practicing beforehand (you'll probably do both). The ordinary keys are comfortably assigned in a logical first-letter-of-a-command manner. There is little logic left when you use the more complicated multi-keystroke operations.

THE BEGINNING

The first thing you'll want to do when the program is on screen is to toggle MIDI to "on" if you have a MIDI device that will actually produce the sound. You may be more comfortable accessing some of the commands from the TitleBar menus at the start, and substituting the keyboard equivalents as you learn and remember them. There are several ways a picture may be "played", either interactively or by computer randomness. At the start, you will no doubt be focusing upon the pictures that are included (some are generation programs that create moving images). In no time at all though (especially if you study the included graphics and their sound capacities) you'll be experimenting with your own visuals as well. Whichever way you choose to have the screen sound out the data, you can also record the musical pattern and replay it later (nice for recording direct to a tape player). The entire pattern can also be saved as a sequence and ported to other software (Dr. T's is mentioned as an example, which means you could also print it out with Dr. T's "Copyist"). Screens can also be saved to disk, and loaded in later.

OPTIONS GALORE

When it comes to applications software, many of us approach it as we do gaming programs,..i.e., if it only does a couple of things (no matter how well), we soon cast it in a dark corner and move on to something else. Given that observation, PIXound will always be in the light, as the options are almost infinite. I'm not going to attempt to delineate everyone here, that would take too much space. But I will touch upon the generalities so that you can appreciate the complexity and variability of this creation.

MIDI users can address output channels and patch bays, so the various harmonies can travel on a chosen path to a specific sound. No reason you couldn't also address other MIDI devices like drum machines, lights, and anything else that can be driven by MIDI signals. Not only do colors relate to sounds, but various saturations of color also manipulate the audible signal. Pastels, for instance, actually sound "lighter", while areas of muddy color sound dark and foreboding. Think of what you can record to videotape in this fashion. Another way to vary the playback is to color cycle the picture, which will cause the sound to speed up as the colors rush by the blitter that senses them. Colors in the palette can actually be "tuned", allowing you to assign various musical attributes to each of them! Harmonies and Rhythms can also be assigned and altered.

There are two functions in PIXound 2.1 that are really mind boggling in terms of allowing you to integrate your own art work. The first, GRAB SCREEN, imports the art from your paint program as it runs in the background and dumps it onto the PIXound screen. I used it with Electronic Arts DPaintIII and it worked fine. The second option is also useful, albeit a bit strange. OVERLAY SCREEN imports your own art screen and blends it with the PIXound screen already visible, thereby abstracting in surprising ways both the visual and the attendant sound. The self explanatory LOAD PIX loads a previously saved IFF graphic from disk.

---

the bottom of the design screen. This adds a new dimension to your riffs, but given just the single instrument voice available with the internal sounds, the results are not great. Harmony is available (see the discussion below) but only with the simple sine, square, and triangular waveforms as voices two and three. It would be much nicer to have a drum sound along with the other single internal IFF instrument voice; to get this effect, you must switch to the use of the MIDI outputs of Hyperchord.

To get a richer sound during my trials, I selected "MIDI" from the menu and listened to the same sequences through my 100-watt stereo system. The sounds of the notes were brief in duration or even staccato when I played a riff at a fast tempo. My experience with the internal Amiga sounds was repeated for MIDI sounds. There are no controls in this program to alter the release or sustain of notes, unlike other algorithmic composition programs I have tried.

You can adjust the parameter settings on the synthesizer's voices (by editing the patch's release parameters), but this is awkward and not under computer control. I used an external analog delay effect device to "stretch" the sounds, but this added more hardware to my setup, and prevented computer control of the voices.

THE PLAY SCREEN

The Play Screen allows real-time control of volume (velocity) and tempo through movement of a set of crosshair lines in the "Vector Play" window vertically and horizontally. The Play Screen also allows real-time control of many other parameters. This screen allows instantaneous selection of mode, rhythm, octave, transposition, and type of harmony. New modes and rhythms can be selected as riffs play, giving the program a good, interactive means for trying, modifying, and fine-tuning a basic riff in real time, without interrupting the flow of music. This allows you to get a set of sounds that best suits your preference.

A problem with the mode and rhythm selection gadgets becomes apparent when you try to create a specific set of modes and patterns. Available modes are represented only as small unnamed squares in a large grid of identical squares. There is no way to label the mode or rhythm associated with each box. It would help to have a scrolling menu to select specific modes and rhythms; or, the boxes could be color-coded or labelled with letters to give the user points of reference.

Other gadgets in the play mode screen allow the user to select several unique and somewhat mysterious effects with the mouse. While I could find no explanations for "superchords", "hyperchords", and "holistic modes" in the accompanying documentation, it was easy to hear the difference that each effect made on a riff. The holistic mode, for example, creates progressions that cycle riffs through several key changes, the keys deriving from notes included in the riffs.

The supertrill is another interesting effect, and when used with MIDI drum voices even gives you the capability to perform good drum rolls effortlessly. For conventional instrument voices like a flute, I don't think a supertrill sounds quite the same as the ones that a skilled musician would play, but this may just be a matter of learning to "play" Hyperchord.

PIXound always lets you know where you're at by giving you echoed data on the TitleBar (Pitch, Scale, Octave, Patches, Sustain toggle, and Cycle toggle). This is not only good, it's vital. Without it, and because of the way that the resident options can complicate matters rather quickly, there's no way you could remember what you did to get where you are. Basically, the "F" keys at the top of your keyboard determine specific modes and scales, from Major/Minor to more esoteric choices ("Gypsy" and "Whole-Tone" scales). I miss having a "Blues" scale option, but maybe that's planned for another revision. The Delete key can be toggled to begin and end the recording of a sequence. From there, it can be saved to disk.

COLOR TRANSFORMATIONS

Since PIXound "plays" your visuals from an assignment of specific note qualities to on-screen colors, it makes sense that there should be global ways to alter the colors, thereby giving you even more options in the audio playback. PIXound allows you to change from one system palette to another (it has eight varieties), or you can create your own palette. Colors can be cycled in any of five ways, and each produces a different harmonic result. The "8" key initiates multi-cycling. Colors can also be reversed and inverted, and the background color can be operated on separately.

CONCLUSION

Have I told you everything? No. Have I attempted to give you a basic idea concerning Pixound's main features, and am I all but guaranteeing you that you will get more then your money's worth from this program? Yes, Yes, Yes! If you are a lover of Amiga generated music, a dabbler in acoustics, an audio scientist, a musical hack, a MIDI enthusiast, a mad scientist, an Amiga visual artist, a just-for-fun kind of person, an adult, a kid, or a seasoned professional musician, buy this program now. If you are none of the above—buy it anyway.

•AC•

PIXound
Hologramophone Research
6225 SW 145 Street
Miami, FL 33158
(305) 252-2661
Price: $99.00
Inquiry #211

HARMONY AND POLYPHONIC, POLYRHYTHMIC RIFFS

Two- and three-part harmonies are easy to generate when playing riffs from the play screen. While only one "melody" or riff can be played at a time, the original note of the riff can be accompanied by one or two copies of the riff, playing in sync with the first riff, but transposed by selected intervals (for example, a fifth and a seventh). Harmony voices can be assigned different MIDI voices, using different MIDI channels and selecting them with the mouse. Different internal Amiga voices, however, can't be played simultaneously.

I attempted to get more than one pattern or riff to play at the same time. What I sought was a way of creating, playing and storing a rhythm line or sequence that was in sync, yet distinct from a melody line. For example, other algorithmic music programs can play one sequence as a rhythm track using drum voices, while a second sequence plays a different ("polyrhythmic") pattern as a bass line, and a third and fourth riff as the lead or melody, using other voices. With Hyperchord, the only polyphonic mode available is the harmony effect. In this mode only one pattern plays, but several different voices can play this pattern simultaneously. The limited polyphony that Hyperchord can produce is able to generate some interesting effects.

Hologramophone people at AmiExpo suggested that I run Hyperchord twice in order to get true polyphonic, polyrhythmic riffs. After the program was running normally, I hit the escape (ESC) key to invoke the "multitasking" option of Hyperchord. Then I went to the workbench screen and clicked on the Hyperchord icon so another copy of the program ran on its own screen. With two programs running simultaneously I played internal Amiga sounds only with one program, and MIDI only with the second. This attempt at polyphonic playing proved to be awkward, since the tempos of the two programs were not in sync. Flipping from one screen to the other was too clumsy to be of much value. I soon abandoned this multitasking mode and tried another approach for polyphonic playing.

A "legitimate" way to get true 2-part polyphonic, polyrhythmic performances from Hyperchord without any 'tricks' is provided on the Play mode screen. A "Play-Along" tool provides a convenient and interesting means for playing a lead or melody line with the mouse. A vertical column of little boxes is on the right side of the vector play area. The column of boxes represents scales of notes covering several octaves. Use the mouse pointer to play notes under complete manual control (as to both the duration and pitch notes). The melody generated with the play-along tool is improvisational, yet easy to keep in key and in sync with the beat.

Using the "swing" tempo, a pentatonic or minor seventh mode, and an acoustic bass plus the drum set from my Korg Symphony module, I created a solid jazz rhythm section. For a lead or melody I added two layered, detuned jazz guitar patches on MIDI channel 5 from my Korg module. This voice was controlled with the play-along tool, letting me create a real-time, improvised, syncopated "modern jazz" piece that sounded realistic and solid. Before using Hyperchord I had never produced a satisfying jazz riff, even using the other Amiga music programs.

CAPTURING RIFFS & PERFORMANCES AS STANDARD MUSIC FILES

A performance could be composed by changing riffs in standard progressions of key changes, melodies modes, tempos, etc. Full performances could be " captured" as an SMUS or a MIDI file in Dr. T's KCS sequencer format. This was easily accomplished by selecting the "record" mode with a single keystroke. I tried this and was able to import a hyperchord performance into Deluxe Music construction Set using the SMUS format. Unfortunately, the melody from the play-along tool was not captured as an SMUS or MIDI file when the program's "record" mode was switched on. This caused the loss of some of the best parts of my performances (the improvised melody line). A riff with one voice could be obtained as a standard MIDI file, using a freely redistributable utility that was on several bulletin board systems. This utility converted a single track saved in Dr. T's KCS format to standard MIDI format. While somewhat inconvenient, this did at least allow me to get one of the features I feel is mandatory for today's music composition software - recording a performance in standard MIDI format.

EXTRA FEATURES

The design and play screens have a multitude of additional features, like continuous play modes, inversion, compres-

sion/expand, MIDI patch selection, etc. These allow more fine tuning of riffs, but take time to learn and understand. Most of these features are described briefly in the program's documentation.

UTILITY PROGRAMS

Several interesting educational/utility programs are on the Hyperchord disk. The Mode Maker and Rhythm Maker programs run separate from Hyperchord, but produce files that can be loaded into Hyperchord later. The Mode Maker allows creation of new modes or scales in an interesting, interactive manner. Graphic images of a set of organ pipes and other objects are presented in this utility. They change to the proper relative lengths when changes to the mode parameters are entered on the screen. This gives the program an intuitive feel which is great for learning music theory, but less effective for algorithmic composition. The Mode Maker program is not able to multitask with Hyperchord (the computer locked up when both were run at the same time).

The two programs complement each other as follows: create a new mode with Mode Maker (at this point you can hear it only as an internal Amiga organ voice). Then save the mode, terminate Mode Maker, and run Hyperchord. Finally, load the newly-created mode into Hyperchord to hear its effect on a particular riff.

Unfortunately, this is just too awkward for interactive composition, unless you create a batch of new modes with the Mode Maker program and store them on a disk. You can then quit Mode Maker, run Hyperchord, and try each of your new modes, switching from one to the other in real time using the Mode grid feature of Hyperchord. This lets you hear and compare how each of these new modes effects the overall sound of a riff, interactively.

RHYTHM MAKER

The rhythm maker is another utility, this one allowing the creation and storage of completely new, customized rhythm patterns for later use with Hyperchord riffs. Using this program, a snare drum sound is generated as an internal Amiga voice forming the basic note or 'beat'. Up to 16 beats can be selected to form a 'rhythm' cycle' or sequence. The process of composing a new rhythm is done with the mouse, first by selecting the number of notes in the pattern, and then assigning a value (duration) to each of the notes. The length of the pattern is chosen by sliding a blue bar above the "rhythm grid". This grid is almost identical to the one at the bottom of the Hyperchord screen. After choosing the pattern's length, the default value for each note can be changed. Changing a note's value is done by selecting it and then clicking the mouse with its pointer positioned in one of the small boxes that forms the top row of the rhythm grid on the screen. For example, the first beat could have its value set as an eighth note, followed by a second beat set equal to a dotted sixteenth note.

When creating a rhythm, a pattern of vertical bars appears over each beat, having a width proportional to the beat's value or duration. Also, the duration of the current beat's delay is shown as a text value on the screen. The small boxes that are used to select the value are unnamed and all have the same color. This makes choosing a specific value difficult, except by trial-and-error. In addition to this problem, another flaw exists. Again, clicking in a tiny box is difficult to do without accidentally choosing the box immediately below the one you wanted to select. If this happens, the entire pattern that you have generated is instantly erased. Since there is no "undo" button, there is no way to recover your work. I

CUSTOMER SUPPORT GRADE: A+

When I first got Hyperchord I found numerous serious bugs in it. This was version 1.1. I called the phone number listed on the registration card for technical support and left a message on an answering machine. To my surprise, within a few hours I got a return call—not to mention lots of helpful advice from one of the program's authors. He told me that all of the serious bugs were corrected in version 1.15, and then answered all of my questions thoroughly. I received an upgrade free of charge soon after sending my original disk to him. He asked my opinion of the program and told me of upcoming improvements to the program, plus additional documentation that was being prepared, including a tutorial. A second call a few weeks later received a similar rapid response. Hologramophone clearly provides some of the best customer support in the Amiga market.

OVERALL IMPRESSIONS

Hyperchord and its accompanying utilities include a variety of features not available in any other Amiga music programs that I know of. Advantages it holds over other programs include the widest range of selectable modes and scales, plus the ability to create and store customized modes. Hyperchord easily produces riffs that no other program can, without the aid of one who is well-trained in advanced music theory. The program therefore is an excellent music theory instructional aid. The riffs that can be produced are not easily edited to generate scores for existing popular or "standard" songs. That is not the purpose of the program. Rather, if you are open to new ideas and experimentation, Hyperchord helps you create riffs of virtually limitless scope and style.

Unlike most other Amiga music software, the program is not copy protected in any way, so the user does not have to go through awkward startup procedures like looking up words in the manual or removing the program disk and inserting a key disk. In addition, the latest version available at the time this was written (version 1.15) is virtually free of bugs to lock up, crash, or ruin work that you have spent hours creating.

Weaknesses in the package include a few minor bugs and problems with the user interface. Entering notes with the mouse is more difficult than in other music composition programs. The documentation is presently too brief (31 pages) and no index is provided. Further, some of the diagrams in the manual are labelled with symbols that do not correspond to the text that references them. A summary sheet, card or on-line help screen should be provided to aid users. This would be helpful even after learning the myriad features of Hyperchord, especially since there are too few menu items to allow selection of these features. More MIDI controls should be provided for operating keyboards and MIDI sound modules.

In general, Hyperchord is worth adding to your existing music software collection. I cannot say that this is definitely the first algorithmic composition program you should buy, since other available programs may have different features of greater significance to you. If you want to learn intermediate and advanced music theory and apply it to the music that you compose, this program is probably your best choice.

•AC•

Hyperchord
Hologramophone Research
6225 SW 145 Street
Miami, FL 33158
(305) 252-2661
Price: $159.00
Inquiry #212

SNAPSHOT

by R. Bradley Andrews

WHERE IN EUROPE IS CARMEN SANDIEGO?

First this month is Where in Europe is Carmen Sandiego?, by Broderbund. This game combines elements of detective work with knowledge of European countries for an interesting gaming experience. It is the third in Broderbund's "Where is Carmen Sandiego?" series, but the first ported to the Amiga (that I know of).

The player takes the role of a new detective of the ACME Detective Agency starting at the rank of Gumshoe. The goal is to find and capture members of Carmen's gang in response to their current crimes in Europe, and thereby advance up the ranks in the agency. From one to four solved cases are required to advance each level. Super Sleuth status is the highest and only happens after the player captures Carmen herself. Game play is fairly simple. Each case begins with a call from the chief, explaining the known details of the crime and the sex of the thief. Some are plausible, such as a stolen gem, but others are very wacky, including stollen mountaintops and geysers.

It is then up to you to follow their path back to the current hideout, a path that will take you through several different countries. At somewhat random times during your pursuit, the chief will fill you in with newly discovered details about the thief, including hair color, eye color, their favorite book, and even their favorite movie. These are important since several characteristics are required to uniquely identify the proper gang member.

Proper identification is necessary to issue a warrant for their arrest. Without the proper warrant, the captured thief will be found not guilty and released from custody.

Two tools are provided to help you in tracking your target. The game package includes a small size Rand McNally Atlas of Europe and an on-screen "computer" is available to identify countries by flag color, monetary unit, and language. One of the game's major claims to fame is its ability to make learning geographical and related facts fun. Just by playing the game and chasing the thieves, any player will gain a bit more knowledge about the European continent.

As a game the program is moderately enjoyable. It can be interesting for the first several pursuits, but extended play can become a bit boring, since the basic mechanics are the same for all play. I was disappointed to find I had to recapture the same crook several crimes later, after they were supposedly put away "for a long time". The graphics look sharp and clear, and many humorous actions are animated with appropriate sound effects. The still frame views that accompany each city location have nicely drawn images representative of the activities each city is famous for.

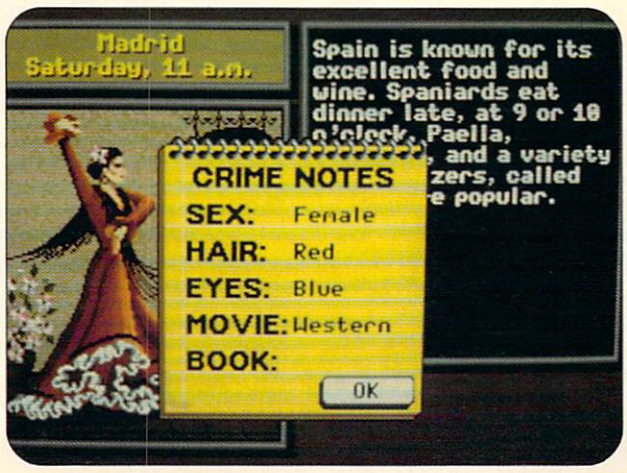
On the whole this is actually a fun game and I would highly recommend it to anyone who would like to learn a bit more about European geography. As long as you don't expect something to match the latest Sierra adventure, you should have a good time.

THIRD COURIER

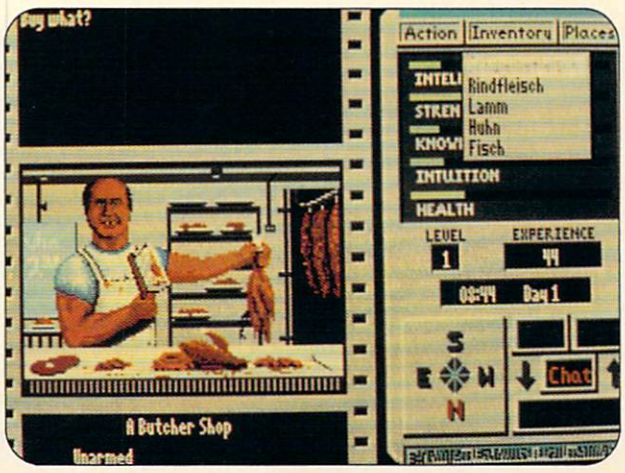
Third Courier is probably the last Cold War game we will see, at least for a while. This game, from Accolade, puts the player in the role of the master spy Moondancer as he attempts to recover stolen NATO defense plans and bring the traitor who caused their loss to justice.

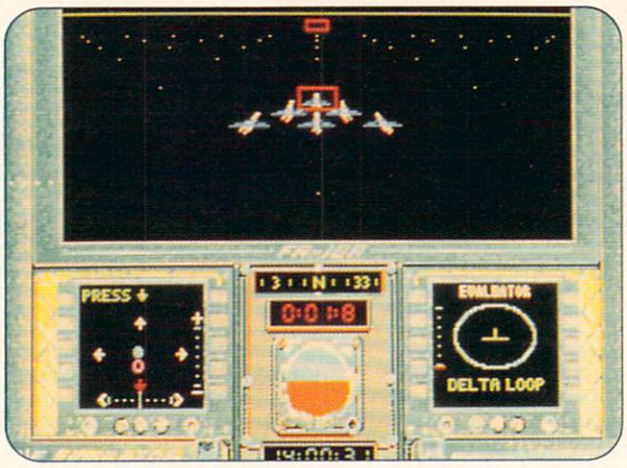
Third Courier is a graphical role-playing game with a similar layout to Uninvited and Shadowgate. The player can move around the discreet areas of the cities of East and West Berlin in an attempt to accomplish the quest. You begin in a small flat rented

Broderbund's Carmen Sandiego Europeon style

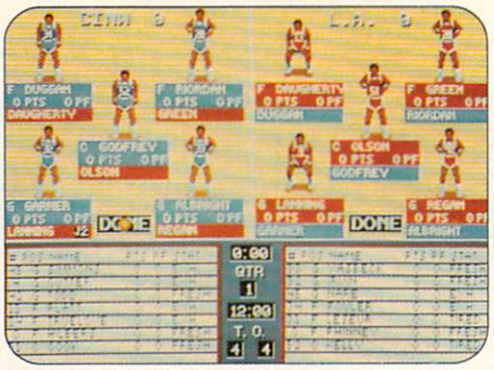


Accolade's Third Courier





Accolade's Blue Angels



Cinemaware's TV Sports Basketball

for you in West Berlin. Your only information about the cities is a small map included with the game. While it does have street names and shows the U-bahn (subway) entrances, it does not have much other use, and it will be your task to explore the city, scouting out the various eating, drinking, living, sleeping and office establishments.

The game play screen is fairly well designed and generally provides the needed information during play. Along the left side is a forward looking view of what is currently in front of your character. The artwork used here is clear and looks nice. The right hand side of the screen holds the player status displays which show health, strength, intelligence, and other important characteristics.

Also on this side are the many action buttons available during play. Either the mouse, the keyboard, or a joystick can be used for play, making it one of the most flexible games around. It was a bit of disappointment that the arrow keys could not be used to move around though, instead the designers chose to use N, S, E, and W to go those directions. It does do one thing in a different fashion than what I have seen in other games. The on-screen "compass", which can be clicked on to move about the city, rotates during movement so the current direction is always on the top. After a brief adjustment, I found this actually worked out well and didn't interfere too much with play.

The game does have what I see as two major flaws. First, it seems West Berlin is just littered with panhandlers and muggers, who are downright hostile and will not hesitate to kill you for the few bucks you have on hand. This would not be so bad, and does add some color to the game, but it seems that you cannot go four blocks without being accosted, and the more encounters, the more likely you are to take damage, and probably even die, before you can drive the freeloader off. A little moderation here would have been appreciated.

The other problem is the somewhat aimlessness of the game. While I did manage to make it to HQ by taking a taxi. (I couldn't walk there since they forgot to list it on the map.) But after stocking up here, I could only wander around the city, beating off muggers. I believe I was supposed to bribe people to get some information, but even after bribing a whole bunch of bartenders and waiters, I found I knew absolutely nothing more than when I started.

There is probably a good game buried in here somewhere. But since I only have a limited tolerance for these types of games anyway, I doubt I will ever dig it out. The presentation of the game is nice, and the interface is good, but with the plot flaws, I cannot really recommend it. If you like starting games where you must figure out virtually everything, you may actually enjoy this game.

BLUE ANGELS

Blue Angels, by Accolade, takes flight simulators one step farther. Instead of simply flying from place to place, or shooting down enemies, the goal here is to do formation acrobatic flying, the kind people all over come out to see the Blue Angels do in Air Shows throughout the country.

Flight Simulation is a popular subject, but it actually looks like Accolade has found a new angle. Not only can you do stunt flying with a single airplane, you can also use the game to fly in a tight formation with four other planes. The game is very flexible at what it does, maneuvers can be practiced in the simulator, at a practice air show, or at the "real" air show.

During action, the camera's view can be taken from many different locations: a chase plane, the cockpit of one of the formation's planes, an observation balloon, or a fixed spot in the stands. As with any flight simulator, it takes a while to become familiar with the controls.

Performing the maneuvers themselves is also challenging. The typical sequence will be to first watch the plane carry out the maneuver while on autopilot. Then you will turn off the autopilot and fly through the square hoops that mark the path for the plane to fly through. Then the same move will be performed without the aids. Then several moves will be integrated into a practice air show, culminating in a "real" air show. The graphics are acceptable. Wire frame images are used for most of the terrain and all buildings, but bit mapped images appear to be used for the planes themselves.

The sound is very simple, adequate, but nothing to get excited about. The game is an adequate flight simulator, but if that's all you want to do with it there are better ones available. But if formation flying appeals to you, check this game out.

TV SPORTS BASKETBALL

Cinemaware has brought out another title in their TV Sports series. TV Sports Basketball brings the ideas in their TV Sports Football to the fast paced game of basketball. TVSB uses the same techniques found throughout Cinemaware games. TVSB focuses on both aspects of the game of basketball. The underlying foundation is based on

player stats and coaching decisions during games. But they do not stop there, a fast-paced arcade-like action game is added to maneuver the actual players around the court during the game. The statistics used are extensive and can be viewed and printed from the clipboard section prior to play. Any team can be viewed from this page, and a wise coach will carefully examine this page prior to playing any important games.

Once the teams have been examined, it is on to the actual game. Games can fall into one of two categories, exhibition or league play. Exhibition games can be between any two teams but do not affect either teams standing in the league. But just playing exhibition games would get old after a while, and sooner or later players will want to go on to a regular season. In regular season games, up to 28 human controlled teams can be played off in one long season. The graphics are up to the standards set by their previous releases. Cinemaware devoted the necessary time to make high quality graphical images that look very nice on the screen and really compliment the game. And the sound is very clear and keyed to the on-screen action, even down to the squeak of the hightops on the court during play.

Each game begins with a smooth animated intro sequence that looks very similar to those used by professional broadcasters covering real professional basketball games. While this is probably not the top limit of what the Amiga can do, it does look very nice and can be enjoyable to watch all by itself. Then each side has the chance to select the starting players to use. Player selection is very important, not only at the start but also during play. Players grow tired and must be rested to obtain maximum performance. But a good coach must also be careful not to pull a player who is "on a roll", even if he is a bit tired, or he may loose a valuable scoring machine.

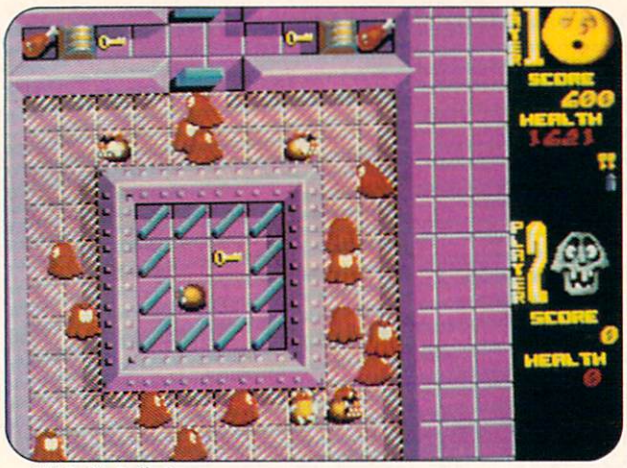
After the starting lineups are selected, it is on to the tip off. Both sides compete for first control of the ball, and from then on it is fast and furious. The box does say that a player can skip the arcade sequences, and just play a more strategic game, but I was unable to locate this feature in the manual or in the game. So be ready for at least some arcade action.

From one to four players can participate in the game action. The first two use the joysticks plugged into the normal slots, while the third and fourth player can only be added if you have a serial port joystick adapter available. But with the adapter, four people, two on each side, can compete at the same time, a feat not many other games can match. In the two joystick mode, both players can either play on the same or opposing sides.

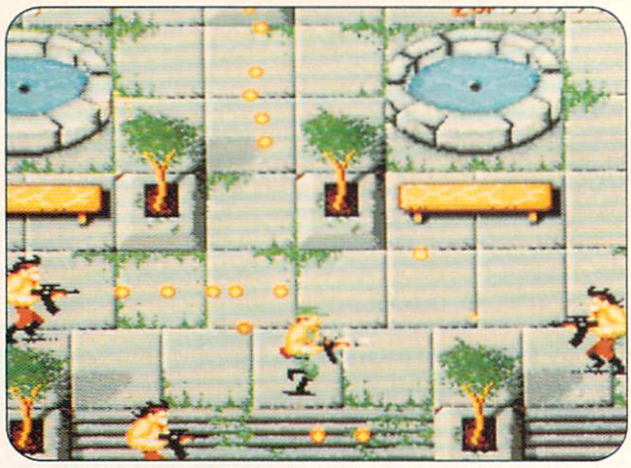
As with other Cinemaware releases, the main problem with the game is its focus on action. It may take a few games to just get the proper motions down. Since this will usually be humiliating defeats, it can be easy to get discouraged and give up. The amount of detail, while great for the simulation, can also be daunting. Since I am not a sports nut, it took me a while to get a grasp on all the statistics, and then to do the necessary replacements, and I am still not where I would like to be. But this game has definite potential, and I would recommend it, especially to those who like TV Sports Football and other Cinemaware releases.

PUFFY'S SAGA

Puffy and his sister Puffyn, two adorable yellow ball shaped creatures,



UBI Soft's Puffy Saga


Virgin Mastertronic's NY Warriors

have been stranded in an alien world. To get back home they must work their way through the 20 levels that make up this bizarre world. Such is the setting in Puffy's Saga, the latest product from UBI Soft, recently imported into the U.S. by Electronic Arts. On a superficial examination, Puffy's Saga looks like a clone of Gauntlet. It features many of the things that made this game so popular: Ghosts, Dragons, dungeon walls, keys, doors, etc. But it is a complete game in its own right and is very fun to play.

The game can be played in one of two basic ways. A player can simply aim at getting through each level as quickly as possible, or he can choose to take his time and explore every nook and cranny of each level, going for maximum points. Each has its own rewards, and the successful player will probably do some of each. The graphics in this game are even better than those in Guantlet. Perspective graphics are used to add a feeling of depth to each level and the items and enemies scattered about each level are very detailed and look very nice. And the on-screen animation is smooth, even when many things are moving at the same time. The sound is also very realistic sounding and enjoyable to hear.

Finishing this game will take a fair bit of playing. While 20 levels may not sound like much, it will take a long time for even fast players to make it to the end of the 20th level. And even more time is possible if you thoroughly explore each level. This game is enjoyable and is worth adding to the collection of anyone who enjoyed Guantlet and similar games.

NY WARRIORS

Finally this month comes another arcade action game, with the focus on the action. NY Warriors was just released by Virgin/Mastertronic and shows that they do know what it takes to make a high quality action game on the Amiga. It seems that the World Trade Center has been overrun by terrorists and only you (and maybe a friend) can save it. But NY has deteriorated a fair bit. Not only do terrorists await you at the end, mean gangs have taken over the streets and you must first fight through many stages of these lethal opponents.

The focus, as with the other arcade conversions by this company, is on the action. Things happen fast in this game. But this fast action is not as much of a problem here as it was in their sports game conversions I covered a few issues back. The graphics are also very sharp and probably the best of this months crop of quality images. The sound is sharp and clear, and most important of all, multiple objects can be on the screen at the same time without any flicker at all. Probably second only to Innerprise's Battle Squadron, I was very impressed with what Virgin/Mastertronic has done with their programming.

Four different difficulty levels are available for play, but all are very fast and the game lacks a truly easy level. The one that is called easy is not that simple and will be challenging for nearly any player. The game also has the problem common with most arcade games that use power-up icons. If you die, you lose your current super-weapon and must start back with a wimpy pea shooter. But special weapons are fairly plentiful, especially in the later levels, so this is not as much of a problem as it might be.

I would have liked the game better if the Easy level had been easy enough to allow me to actually complete the game after a bit of play. But even with the difficult here, I found I could actually do well after I learned where things were and how to best eliminate foes. This is a worthwhile purchase for nearly any arcade action fan. The combination of quick action play, excellent graphics, and top notch sound make it a good purchase for everyone's arcade action library.

•AC•

Products Mentioned

Where in Europe is Carmen Sandiego?
Broderbund Software Inc.
17 Paul Drive
San Rafael, CA 94903
(800)521-6263
Price - $44.95
Inquiry #202

The Third Courier
Accolade
550 South Winchester Boulevard, Suite 200
San Jose, CA 95128
Price - $49.95
Inquiry #203

Blue Angels
Accolade
550 South Winchester Boulevard, Suite 200
San Jose, CA 95128
Price - $49.95
Inquiry #204

TV Sports Basketball
Cinemaware Corporation
P.O. Box 5083
Westlake Village, CA 91359
Price - $49.95
Inquiry #205

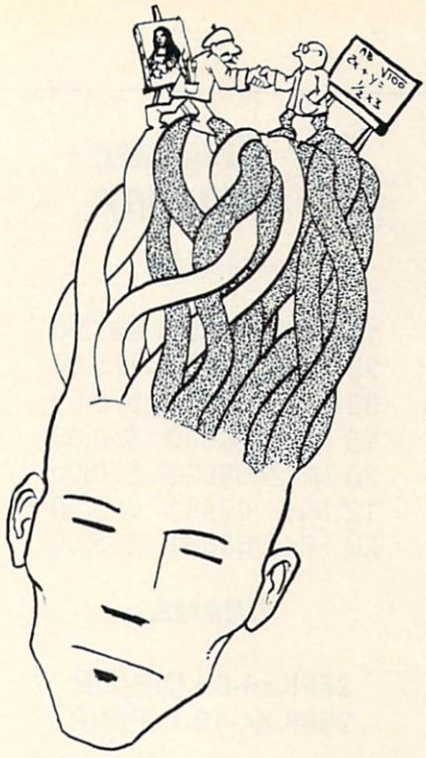
Puffy's Saga
UBI Soft
Electronic Arts Distribution
1810 Gateway Drive
San Mateo, CA 94404
(415)571-7171
Price - $34.95
Inquiry #206

NY Warriors
Virgin/Mastertronic
18001 Cowan, Suites A & B
Irvine, CA 92714
(714)833-8710
Price - $49.99
Inquiry #207

Synchronicity:

Right & Left Brain Lateralization

by John Iovine



PEOPLE ARE TRADING IN THEIR TRANSCENDENTAL MEDITATION mantras and alpha brain training devices for the latest in consciousness-expanding techniques: right and left brain synchronization using sound, music and light.

BRAIN SCIENCE 101

Brain researchers have determined that people who use both sides of their brain equally usually have enhanced creativity and problem solving abilities.

When you look at the human brain, it is easy to see that, in reality, it is a double organ, constructed of symmetrical and identical hemispheres.

Information gleaned from the last 40 years of brain research shows that each hemisphere of the brain has its own methodology of problem solving ability and its own way of perceiving the world around it. The right brain is non-verbal, emotional, holistic and spatially oriented (ie. the emotional side). The left brain is verbal, sequential, literal and emotionally flat. (the logical side.) The reverse is true for 30 percent of left handers.

Both sides of our brain are connected by millions of nerve fibers called the "corpus callosum". This brain organ is responsible for exchanging information between the right and left brain hemispheres.

Just to give a little background on this specific area of brain research, much of the information gained from this type of research began with operations on people who were experiencing severe epileptic seizures. It appears that the onset of a seizure begins with a localized abnormal electrical activity in the brain that quickly spreads throughout the brain. The doctors decided to cut through the corpus callosum, in effect separating the hemispheres, in an effort to keep the seizures localized.

The operation succeeded, but left the patient with two distinct split brain personalities. Other information was obtained from people who had suffered strokes (cerebral hemorrhages) that destroyed one half of the brain while leaving the other intact.

Fortunately today, researchers don't have to experiment with anyone who has gone through this kind of tragedy. Since each half of the brain is fed by a different artery in the neck, researchers can selectively put one side of the brain asleep using a tranquilizer injected into one of the arteries. This is called the "Wada procedure."

RIGHTY OR LEFTY?

When presented with a problem one side of the brain usually takes over and becomes the dominant problem solver, depending upon the nature of the problem. A mathematical or verbal problem will usually be handled by the left hemisphere. The right hemisphere will take over in visual and spatial problems. All this hemisphere switching happens below our level of awareness (subconsciously).

Researchers David Galin and Robert Ornstein first discovered this division of labor in the brain in 1972. They recorded EEG (electroencephalograms) patterns separately from both

brain hemispheres. When various problems were presented to the subject they observed that one hemisphere became the dominant problem solver depending upon the nature of the problem. When a verbal task was assigned to the subject, a decrease in the alpha rhythm was noticed on the left side while it remained constant on the right. This clearly indicated that the left brain was working on the problem while the right brain continued to idle. When a visual task was assigned to the subject, the opposite results were observed.

In some instances both halves of the brain compete for control; this happens when both sides want to answer a particular problem or question. This can result in stammering and stuttering.

FREUD

The right brain is strikingly similar to what Sigmund Freud described as the subconscious (unconscious) mind. Many techniques used by psychologists to probe a patient use right brain superiority in task handling. The Rorschach ink blot test, for example, where an ink blot is presented to a patient for image association, is clearly a right brain task.

Another Freudian concept used for psychological analysis, dreams are strongly located in the right hemisphere.

These approaches work because the right hemisphere has its own memory of events, and they are not necessarily the same memories as the left hemisphere's. Repressed memories and traumatic events from a patient's past may be brought to the surface by employing these psychological tools.

In consideration for what we have learned on right and left brain hemispheres, I think it would be appropriate if we renamed the subconscious or unconscious right hemisphere "non-verbal". Since the right brain is equally conscious, and we don't want to unjustly insult ourselves, do we?

BRAIN WAVES

0 Hz	Brain Dead
Delta 1-4 Hz	slow waves more common in children and a normal part of their development. Adults produce delta brain waves from time to time during sleep.
Theta 4-7 Hz	appears to be related to problem solving, sorting and filing of information within the brain's memory. Theta waves are also produced by Zen practitioners in deep meditation.
Alpha 8-13 Hz	dominant rhythm in normal adult EEG when subject is relaxed, awake with eyes closed.
Beta 14 Hz+	appears in normal adults who are "alert" as opposed to relaxed. Being in "beta" is identified as being tense, irritable and basically unpleasant.

NEW WAVE

In the 1960's and early 1970's Transcendental Meditation and bio-feedback devices to help produce alpha waves became something of a rage. It promised enlightenment, relaxation and stress reduction. Today there is a growing interest in right and left brain synchronization.

This technique discovered by Robert Monroe promises to put anyone into alpha, theta or delta states (beta state is the norm) by listening to sound that has a synchronized beat to it. The brain wave pattern becomes entrained by the synchronized beat and follows it. The synchronized beat should be at the EEG frequency one is interested in obtaining. For example, you might try a 9 Hz beat frequency for alpha, 6 Hz for theta and 4 Hz for delta.

MIND GYMS AND MIND MACHINES

Mind machines and mind gyms are offered as a quick fix for everyday stress. Some claim to train your mind to be more creative, productive and imaginative. More ambitious advertisers claim physical benefits such as lower blood pressure, alleviating migraine headaches, and improved intellectual capability. The improved intelligence is derived by maximizing the lateralization of the right and left hemispheres for problem solving.

Mind machines operate on the premise that your state of mind can be influenced by exposure to sound, light and electromagnetic fields. Mind machines such as the Synchro-Energizer are for sale being advertised in many magazines like *Omni* and *Psychology Today*. These machines typically consist of goggles with flashing lights and headphones that play synchronized sound. The light flashes from the goggles are synchronized with the beat frequency of the sound from the headphones.

Mind gyms are turning up around the country. In these "gyms" you have basically the same type of equipment wired to handle a multitude of customers.

WITH A GRAIN OF SALT

All claims should be taken with a grain of salt. I don't think any of these companies have an FDA approval or have filed for such

approval. Fortunately for us, we don't have to buy any machine or invest a large sum of money to start checking these claims out for ourselves. We have an Amiga computer that is quite capable of imitating and possibly surpassing these stand alone sound and light machines. But we'll come back to this later on.

FDA

It is the FDA (Food and Drug Administration) responsibility to keep worthless or potentially hazardous devices out of the public's hand. Any device that could be classified as a "medical device" must go through clinical trials before being marketed. Are these sound and light machines medical devices? A lot depends upon the wording of the advertisement. But the claims made by many of these companies are really pushing the issue.

The reason why these machines are allowed on the market in the first place without clinical trails is an FDA loop hole in the 510-K statue. This loop hole waives the standard clinical trials for any device that has been on the market before 1976. A proto-type sound and light machine called ISIS has been around and marketed since 1971.

SOUND

Sound at the brain EEG frequencies is far too low to hear. But by playing two sounds together whose frequencies vary by a small amount, (as an example let's use 9 Hz, alpha freq.), a beat frequency of 9 Hz can be heard. This sounds like a wah-wah-wah or wavering in and out of the sound frequency and volume. What you're hearing is actually the difference between the two frequencies.

A sound example is worth a thousand words. In order to clarify this explanation, please power up your Amiga computer and load AmigaBASIC. In the AmigaBASIC window enter:

```
Rem Sound Test
Rem Voice 0 & 3 are Left Channel
Rem Voice 1 & 2 are Right Channel
Sound Wait
Sound 523,70,255,0
Sound Resume
```

Run this program. What you hear is a "C" note. Now enter this additional line after the first sound statement and before the sound resume statement.

```
Sound 532,70,255,1
```

Run the program again. Notice the difference; you should hear the note wavering in and out. That's the beat frequency, the difference of 9 Hz between both sounds.

FFR FREQUENCY AND FOLLOWING RESPONSE

This is a term constructed by Monroe to describe his technique. Essentially by presenting these sounds separately to each ear, the EEG wave pattern of the brain will follow the beat frequency. So if we used the program example above, this should make our brain wave resonate at 9 Hz bringing on an alpha state.

It is necessary and important to use stereophonic headphones connected to the right and left channels on the Amiga audio out. The sound must be mixed intra-cranially in order to generate any effect.

I should point out that when scientists originally tested experienced Zen meditators, their right and left hemispheres did fall into synchronization.

THE PROGRAM

This program is for those of you who would like to try this technique. You can attempt enhanced learning, programming, relaxation or whatever.

The program first queries you for what frequency you'd like to try; alpha, theta or delta. Then you will be asked for a time period for how long the program will produce the sound. Once the program is started, you can use your gadget to shrink the window and then open another to work on something else.

MUSIC

Listening to a monotone note can get boring. There isn't any reason I can think of that would prevent this system from working with music. The basic idea of course would be to assign one channel to follow the other with a frequency difference of the brain wave state you'd like to explore.

CONSTRUCTION

There isn't any construction involved in this project. You only need two pieces of equipment. One is stereo-headphones, the other is a "Y" adapter. The "Y" adapter must have 2 RCA phono plugs on one end that plugs into the two audio-out sockets on the Amiga. The other end of the adapter must have a socket to plug your headphones into. Plug the "Y" adapter into the Amiga, and the headphones to the "Y" adapter and you're ready to go.

Headphones and the "Y" adapter can be purchased at a local Radio Shack or Stereo store.

Test the unit out by putting on the headphones and playing a sound first in one channel and then the other. You should hear

the sound on only one side depending upon which channel is active.

CONCLUSION

We have only scratched the surface of the current happenings in brain research. I do not have an EEG machine to verify whether this technique actually works. I do plan to build one later this year as an interfacing project for the Amiga computer. In addition if there is sufficient reader interest in doing so, I will write another article and a circuit that adds the flashing lights to go along with the synchronized sound section. Or perhaps a simple test program and article to determine your own right- and left- brain lateralization.

It has been said and written many times that humans only utilize10 percent of their brain capacity. In as much as there are so many claims that these devices improve your intelligence and allows one to utilize more than the standard 10 percent of the brains capacity, I want you all to know that I use at least 10 percent of my brain or 10 neural synapse firing (whatever comes first), whenever I write, whether I need them or not.

Listing

```
REM Sound Synchronization Software
REM By John Iovine

REM Channel 0 & 3 are Left
REM Channel 1 & 2 are Right
mmenu:
CLS:LOCATE 7,25
PRINT  "Menu"
PRINT " 1) Set EEG Frequency "
PRINT " 2) Set Time"
PRINT " 3) Run"
PRINT " 4) Quit"
PRINT:PRINT "Enter Selection (1-4)"
  INPUT a
    ON a GOTO EEG,ptime,start,pEND

  EEG:
  CLS:LOCATE 7,30
  PRINT  " EEG Menu"
  PRINT
  PRINT "1)  Delta "
  PRINT "2)  Theta"
  PRINT "3)  Alpha"
  PRINT "4)  Beta"
   PRINT :PRINT "Enter Choice (1-4)"
    INPUT a
      IF a=1 THEN b=3
      IF a=2 THEN b=6
      IF a=3 THEN b=9
      IF a=4 THEN b=12
    IF a<1 OR a>4 THEN EEG
    GOTO mmenu
ptime:
  CLS:LOCATE 7,30: PRINT "Set Time Elapse"
  PRINT
  PRINT "Enter number of minites program to run."
  INPUT t
  IF t<0 THEN ptime
  GOTO mmenu
start:
CLS:LOCATE 7,7:
PRINT " At this point you may shrink this window"
PRINT "using the gadget in the lower right hand corner"
PRINT "and open another window. Or use the back gadget "
PRINT "in the upper right to get back to an opened window"
 IF b=0 THEN b=9 :REM default to alpha
 st = t*60
 stimer = TIMER + st


     WHILE TIMER < stimer
 sd1=638.25: REM 1st note value
   SOUND WAIT
 SOUND sd1,77,255,2: SOUND sd1+b,77,255,3
   SOUND RESUME
    WEND

GOTO mmenu
pEND:
CLS:END
```

•AC•

AmiEXPO '90
Basel, Switzerland

The Swiss really know how to throw a party for the Amiga. And why not? It's Europe's number one selling machine.

by Peter Sacks

ABOUT 14,000 PEOPLE ATTENDED the AmiEXPO in Basel, Switzerland. Held on May 9 through May 12, the show gave Europeans some fascinating glimpses into what lies ahead for the Amiga in the 1990s.

The good folks at Commodore Switzerland had the honor of officially introducing the flashy new A3000 to the European public at the recent AMIGA '90 Basel show.

While the A3000 was the main attraction for most visitors, there were also several other interesting products highlighted there. It was a good mixture of attendees: in Europe—where there are about 500,000 Amigas in use—the Amiga is mainly a game machine for youths and normally those young people are the most conspicuous crowd at shows.

Not so at Basel, which was a thoroughly professional show with a myriad of interesting and well-informed people in attendance. Though not quite on par with Cologne in November '89 (40,000 attendees forced exhibitors to close the doors sometimes because the hall was too overcrowded), it was for the most part a highly successful show.

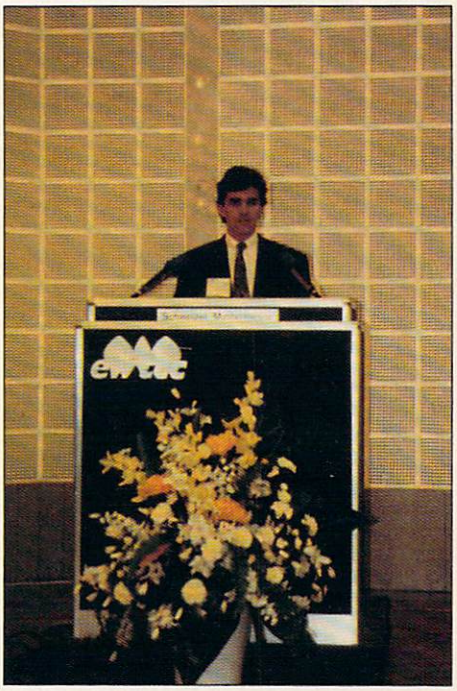
So what of note took place at Basel?

Commodore's Dave Hainey, one of the main designers of the A3000, was extremely helpful in answering hundreds of questions. Commodore sold assorted promotional gifts and debuted their new multimedia program—AmigaVision—right alongside the spectacular new A3000.

Alexander Gloss, President of AmiEXPO Inc., gave an interesting speech on the history of AmiShows Europe, and discussed shows planned for the future. Among them, an AmiEXPO is planned for London in 1991. Ralf Hollax, General Manager of AmiShows Europe, focused on show organization matters and assured those of us in Basel that AmiShows had learned a lot from 1989's oversuccess in Cologne. They have moved to expand exhibition space to 25,000 square meters for this year's show in Cologne, so we will surely have a better go of it this time.

Other speeches were given by August Harder, a Swiss Commodore manager and Wolfram Hoefler of Markt & Technik (the patronaging publishing house) also talked about the incredible success of AmiEXPO in Cologne. Albert Absmeier, chief editor of Amiga Magazin (an official co-patron of AmiEXPO's in Europe, along with Commodore), said "Informing before buying is important—an exhibition is the ideal forum for information-gathering". Well-known Amiga artist and video designer Joel Turner showed a nice film he made with the Amiga.

As for new products exhibited at Basel, we had a look at an interesting product named the 'Colourbox,' a blue-box system for the Amiga presented by Intelligent Memory. Just in case you are not →



***Above:** Alexander Gloss, President of AMI Shows, Inc.*

***Left:** Commodore shows off the A3000.*

The Power Is Amazing.

Only Amazing Computing for the Original Amiga gives you *the power to do more* with your Amiga!

In recent issues, ***AC*** has given you the power to turn your A1000 into a ROM-based machine.

To upgrade your A500 to 1 megabyte.

To transform your Amiga into a biofeedback/lie detector device.

To use your Amiga as the command center for a remote-controlled home of the future.

A future that is *in your hands now*, thanks to ***Amazing Computing***!

Subscribe Today!

There's power in our new product previews, product reviews and tutorials, too. Plus, ***AC*** takes you to all the major shows with detailed, timely coverage!

COMPARE! ***AC*** actually publishes many more pages of solid information – insightful columns, departments and features – than *any other Amiga publication!*

NOW – A LIMITED TIME OFFER:
Put a full year of **AC POWER** in your hands for just **$24**!
Or, subscribe for 2 years and pay just **$38** –
that's a **SAVINGS** of ***60% off the newsstand price!***

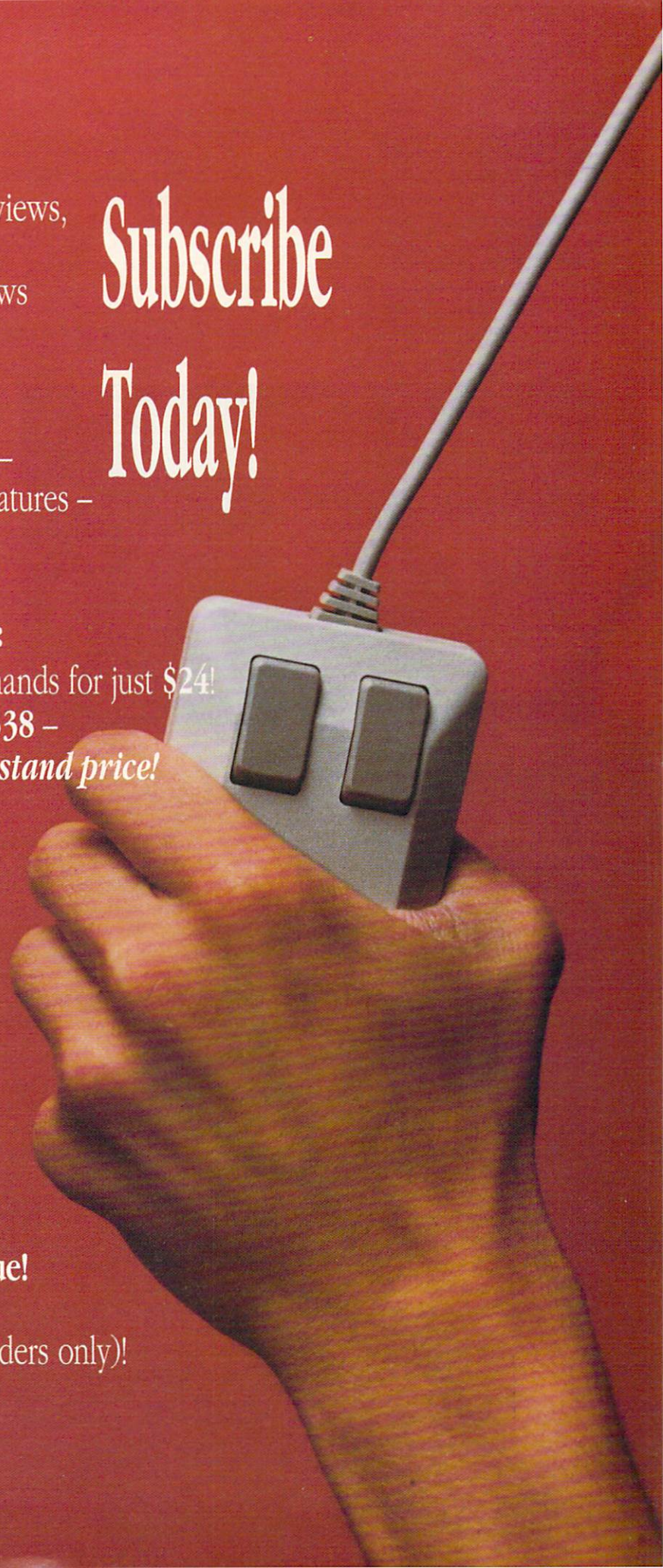
Remember – we also place
the entire Amiga marketplace
at your fingertips in the pages of

AC's GUIDE TO AMIGA

It's the *only* publication in the world that lists *every product* currently available for the Amiga!

To subscribe, just use one of the cards provided in this powerful issue!

Or, call **1-800-345-3360** (credit card orders only)!



familiar with what a blue-box is, it's a piece of video equipment that takes two video sources (one of which includes a blue field, normally in the background), and superimposes one shot on top of the other by displaying the field of view from one source in place of that field of blue in the second. With this technique, Superman is able to fly through the streets of New York, and with your Amiga and the Colourbox, you could do the same.

GVP's booth

The difference between a normal blue-box and the Colourbox is that the Colourbox allows you to use **any** color for the "blue" purpose. Of course, you can also use it as a normal Genlock and fade pictures in and out. The Videobandwidth is over 5 MHz. The Colourbox will be shipping in October 1990 and is priced in Germany about DM 1800—that would be slightly over $1000 U.S.

Several exhibitors presented hard drive solutions for the Amiga. One of the most long-awaited and fascinating of those was Ariba, an internal 20 Meg hard drive from Gigatron for the Amiga 500. Ariba consists of a controller, which is plugged directly into the base of the 68000, and a 2-1/2 inch, 23-millisecond drive. The combination has a transfer rate of about 300 KByte per second, and will be sold in Germany for DM 1298, which is about $750.

IVS, represented in Europe by DSP, showed their new Trumpcard Professional, a SCSI-Controller for the Amiga with incredible transfer rates. The normal version is able to achieve rates of 900 KBytes per second. Combined with a 68030 it may reach a tremendous 1.5 MBytes/second—if the hard drive is able to go along. At the show there was a nice demo in which an Amiga played a Disney film of about 3 minutes in length directly from the hard drive, with 30 HAM pictures per second, each about 45 kilobytes. Unfortunately, I was unable to get any comments on price and availability.

Combitec, a West German company, was some time ago able to finish their Atari ST-Emulator for the Amiga, but shortly afterwards they went bankrupt. As it was a good working piece of soft/hardware (it is a module for the A2000 and an emulator of 14k length only), the rights were directly bought by 3-State. They exhibited it at the Basel show and are going to sell the soft/hardware for around DM 600 ($340 U.S.). Unfortunately, I'm not sure of the eventual market for it, as several PD versions of an ST-Emulator are now being passed around in Germany, and they do not need any hardware. Their only fault is that they are incapable of writing ST disks, but they can read them without any problems.

bsc, a German third-party supplier, showed some really "amazing" new products for the Amiga. The most interesting of these was the NonFlicker Cable, a simple cable to connect a cheap PC TTL-Monitor to the Amiga. The cable is able to reduce flickering in interlace mode, but it is only capable of displaying four colors, so you won't have too much fun with color-intensive programs. With the cable, your monitor will still flicker a bit, but it is a really good and most economical (only about $40 U.S.) alternative to the expensive Flicker-Fixer. Also shown by bsc was Ultradesign, a good-looking CAD program, which seems very easy to use as it is completely mouse-driven and appears quite powerful. But we should have a closer look at it before commenting too fully. What is not so amazing is the price, around $600.

Memory and Storage Technology (M.A.S.T.) started business in Germany just a short time ago, and had a big booth at the Basel show. They had good success with their existing product line at the show, and also showed off their brand-new Blitz-Basic!, a basic-like language which gives much more control over graphics, sprites and sound than AmigaBASIC gives to the programmer. It lets one manage the copper and blitter, and provides for dual playfields, double buffering and smooth scrolling. With it, M.A.S.T. also supplies a compiler which produces pure assembler code, and doesn't use the slow Amiga OS Libraries. The editor with Blitz-Basic! is also excellent, and long-suffering AmigaBasic people will especially appreciate that it not only has the functions of a small text editor, but also has online help texts for keywords. A brief overlook of Blitz-Basic! proves to be very good; in particular, the demos were impressive, as were the short and neat programs used to create them. It appears that for around $100 we are getting an excellent piece of software.

Gold Vision, a German company headquartered in Berlin, presented their "IFF to Vector" program Vector-Trace. Vector-Trace turns a normal IFF image into a vectorized version, so you can blow it up without getting the so-called Jaggies. This is extremely useful for DTP programs, especially when working with scanned images or logos. VectorTrace can save the vectorized image in AegisDraw, PostScript or VideoScape3D format so that you have professional-looking images in DTP form, even if you had to enlarge the image. VectorTrace is sold in Germany for DM 150, (about 85 dollars U.S.).

A neat product came from Rossmoeller. Their A500 Power PC Board is a PC card plugging into the internal expansion slot. The board is fitted with 1 Meg of RAM and a NEC V30 processor with 8 MHz, and supports Hercules and CGA graphics. In the PC mode it has 768K free RAM, in the Amiga mode you can use it as normal expansion memory with 512 K autoconfiguring RAM and a 512K RAM disk. Interestingly, the board uses the normal Amiga ports, so you can use your drives/mouse/joystick and parallel/serial connectors of the PC. The complete board with MS-DOS 4.01 sells in Germany for DM 798, or $450.

When looking back on the show, one has to say that for the attendees it was a very enlightening and successful show. One was able to meet and talk to many knowledgeable Amiga professionals about their experiences with this product or that, and to exchange some hints.

•AC•

List of Exhibitors

3-State Computertechnik
Schaumburgstrasse 17
4350 Recklinghausen
West Germany
Telephone: 49-2361-16207
Inquiry #213

A + L AG
Daederiz 61
CH-2540 Grenchen
Switzerland
Telephone: 41-65-520311
Inquiry #214

Alcomp
Giescher Weg 22
5012 Bedburg
West Germany
Telephone: 49-2272-2093
Inquiry #215

AMIGA-Magazin
Verlag Markt & Technik
Hans-Pinsel-Strasse 2
8013 Haar bei München
West Germany
Telephone: 49-89-46130
Inquiry #216

bsc Büroautomation GmbH
Schleissheimer Strasse 205a
8000 München 40
West Germany
Telephone: 49-89-3084152
Inquiry #217

Manfred Carle Hard-Software
Langstrasse 23
6450 Hanau
West Germany
Telephone: 49-6181-251628
Inquiry #218

Commodore Büromasch GmbH
Langenhagstrasse 1
CH-4147 Aesch
Switzerland
Telephone: 41-61-707111
Inquiry #219

Comp-U-Save
414 Maple Avenue
Westbury, NY, USA 11590
Telephone: 516-997-6707
Inquiry #220

Compulit Data Service
Heinstrasse 23a
6368 Bad Vilbel
West Germany
Telephone: 49-69-844819
Inquiry #221

Demonware
Strahlenberger Strassse 125a
6050 Offenbach
West Germany
Telephone: 49-69-8004703
Inquiry #222

Donau-Soft
Postfach 1401
8858 Neuburg/Donau
West Germany
Telephone: 49-8431-49798
Inquiry #223

DSP Hard & Software
Schaufelweg 111
CH-3098 Schliern
Switzerland
Telephone: 41-31-535351
Inquiry #224

DTM Werbung & EDV GmbH
Poststrasse 25
6200 Wiesbaden
West Germany
Telephone: 49-6121-502059
Inquiry #225

Dynamic Computer
Gutenbergstrasse 5
CH-1023 Genf
Switzerland
Telephone: 41-22-444017
Inquiry #226

Elepro AG
Furtbachweg 63-65
CH-8304 Wallisellen
Switzerland
Telephone: 41-1-8302000
Inquiry #227

FSE-Computersysteme
Schmiedstrasse 11
6750 Kaiserslautern
West Germany
Telephone: 49-631-67096
Inquiry #228

Gigatron
Resthauser Strasse 128
4590 Cloppenburg
West Germany
Telephone: 49-4471-3070
Inquiry #229

Gold Vision
Kurfürstendamm 64-65
1000 Berlin 15
West Germany
Telephone: 49-30-8833505
Inquiry #230

Great Valley Products, Inc.
225 Plank Road
Paoli, PA, USA 19301
Telephone: 215-889-9411
Inquiry #231

Heutronic AG
Unterführungsstasse. 29
CH-4601 Olten
Switzerland
Telephone: 41-62-260222
Inquiry #232

ICD Europe GmbH
Am Goldberg 9
6056 Heusenstamm
West Germany
Telephone: 49-6104-6403
Inquiry #233

Intelligent Memory
Waechtersbacher Strasse 89
6000 Frankfurt 61
West Germany
Telephone: 49-69-410072
Inquiry #234

Irsee-Soft
Gruentenstrasse 6
8951 Irsee
West Germany
Telephone: 49-8341-8211
Inquiry #235

Konyo Electronics Vertriebs GmbH
Auwiesenweg 3
8049 Unterbruck-Fahrenzhausen
West Germany
Telephone: 49-8133-801
Inquiry #236

Kupke Computertechnik GmbH
Burgweg 52
4600 Dortmund 1
West Germany
Telephone: 49-231-818325
Inquiry #237

Logico Software
5, Ch. des Paleyres
CH-1006 Lausanne
Switzerland
Telephone: 41-21-265212
Inquiry #238

Logo Software
Haferfeldstrasse 38
8901 Meitingen
West Germany
Inquiry #239

Masoboshi
Joachimstrasse 16
4630 Bochum
West Germany
Telephone: 49-234-308151
Inquiry #240

M.A.S.T. GmbH
Theodor-Heuss-Ring 19-21
5000 Koeln 1
West Germany
Telephone: 49-221-7710918
Inquiry #241

M.A.S.T. USA
1395 Greg Street
Sparks, NV, USA 89431
Telephone: 702-359-0444
Inquiry #242

Microtron Computerprodukte
Bahnhofstrasse 2
CH-2542 Pieterlen
Switzerland
Telephone: 41-32-872429
Inquiry #243

Micro-Systems Software MSS
12798 Forest Hill Blvd, Suite 202
Palm Beach, FL, USA 33414
Telephone: 407-790-0770
Inquiry #244

Miky Wenngatz
Jaegerweg 31
8031 Gilching
West Germany
Telephone: 49-8105-24540
Inquiry #245

Mindware International
230 Bayview Dr., Suite 1
Barrie, Ontario, Canada L4N 4Y8
Telephone: 705-737-5998
Inquiry #246

Otronic Computer und
Bauteile Shop Handelsges.m.b.H.
Bleibtruestrasse 2/1
A-1110 Wien
Austria
Telephone: 43-222-767001
Inquiry #247

Print-Technik GmbH
Nicolaistrasse 2
8000 München 40
West Germany
Telephone: 49-89-368197
Inquiry #248

Reisware
Postfach 36
5584 Bullay
West Germany
Telephone: 49-6542-2086
Inquiry #249

Rossmoeller GmbH
Neuer Markt 21
5309 Meckenheim
West Germany
Telephone: 49-2225-2061
Inquiry #250

Softlogic Publishing Corp.
11131 F.S. Towne Sq.
St. Louis, MO, USA 63123
Telephone: 314-894-8608
Inquiry #251

Softwareland AG
Franklinstrasse 27
CH-8050 Zürich
Switzerland
Telephone: 41-1-3115959
Inquiry #252

Schneider Verlag
Am Weinberg 46
8301 Arth
West Germany
Telephone: 49-8704-1597
Inquiry #253

Supra Corporation
1133 Commercial Way
Albany, OR, USA 97321
Telephone: 503-967-9075
Inquiry #254

Telekommunikation Kaaben-Riis Gbr.
Projensdorfer Strasse 14
2300 Kiel 1
West Germany
Telephone: 49-431-337881
Inquiry #255

TSS Handic Plastics
Elementenweg 18/C
NL-3201 LG Spijkenisse
Netherlands
Telephone: 31-1880-22220
Inquiry #256

Videocomp
Berner Strasse 17
6000 Frankfurt/Main 56
West Germany
Telephone: 49-69-5076969
Inquiry #257

Vidtech International
2822 NW 79 Avenue
Miami, FL, USA 33122
Inquiry #258

Vortex Computersysteme
Falterstrasse 51-53
7101 Flein
West Germany
Telephone: 49-7131-50880
Inquiry #259

Weka-Verlag
Hermetschloo 77
CH-8010 Zürich
Switzerland
Telephone: 41-14-328432
Inquiry #260

X-Pert Computer Service GmbH
Weiherwiese 27
6270 Idstein
West Germany
Telephone: 49-6126-8809
Inquiry #270

MONTGOMERY GRANT

ESTABLISHED 1967

AMAZING COMPUTING 7/90

OUTSIDE USA & CANADA CALL
(718) 692-0790
FOR CUSTOMER SERVICE
Call: Mon-Thurs, 9AM-5PM
Fri, 9AM-4:30PM (718) 692-1148

Retail Outlet, Penn Station, Main Concourse
(Beneath Madison Square Garden) NYC, N.Y., 10001
Store Hours Mon-Thurs 9-7:30/Fri 9:00-6:00/Sat CLOSED
Sun 9:30AM-7PM

FOR ORDERS & INFORMATION IN USA & CANADA CALL TOLL FREE
1-800-759-6565

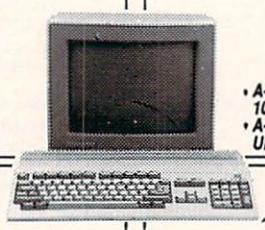
OR WRITE TO:
Montgomery Grant; Mail Order Department P.O. Box 58, Brooklyn N.Y., 11230
FAX NO. #7186923372
TELEX 422132 MGRANT

ORDER HOURS: Mon-Thurs, 9:00am-7:00pm / Fri, 9:00am-6:00pm / Sat CLOSED/Sun 9:30am-6:00pm (ET)
NO SURCHARGE FOR CREDIT CARD ORDERS / WE INVITE CORPORATE AND EDUCATIONAL CUSTOMERS
RUSH SERVICE AVAILABLE / TOLL-FREE TECHNICAL SUPPORT

AMIGA 500

- Amiga 500 w/512K RAM
- Built-in 3.5" Disk Drive
- 40MB Hard Drive
- Mouse
- System Software
- Amiga Basic

$519



AMIGA-500 COMPLETE w/1084 RGB COLOR MON.
$789

- *A-500 w/1084 & 1010 DISK DRIVE* **$939**
- *A-500, 1084 & 512K UPGRADE (1MB Total)* **$875**

AMIGA 2000

- 1MB Expandable to 9MB
- Built-in 3.5" Disk Drive
- Mouse
- System Software
- Amiga Basic

$1249

AMIGA-2000 w/1084 RGB COLOR MON.
$1529

NEW **AMIGA-3000**
CALL

AMIGA 500 RGB COLOR PACKAGE

- Amiga 500 w/512K RAM
- Built-in 3.5" Disk Drive
- Mouse
- RGB Color Monitor
- System Software
- Amiga Basic

$699

AMIGA 2500/30

- 3MB RAM
- 25 MHz.
- Built-in 3.5" Disk Drive
- 40MB Hard Drive
- Mouse
- System Software
- Amiga Basic

$3299

AMIGA 2000 RGB COLOR PACKAGE

- Amiga 2000 Computer
- 3.5" Disk Drive
- Mouse
- RGB Color Monitor
- System Software

$1429

AMIGA 2000 H.D.

- 1MB RAM
- Built-in 3.5" Disk Drive
- 40MB Hard Drive
- Mouse
- System Software
- Amiga Basic

$1799

ADDED BONUS: WE OFFER 1 YEAR PARTS AND LABOR WARRANTY ON ALL AMIGA COMPUTERS PURCHASED THROUGH MONTGOMERY GRANT!

The Lowest Pricing - Lifetime Toll Free Technical Sopport - Extended Warranty STANDARD - All You Expect From MONTGOMERY GRANT

WE WILL MEET OR BEAT ANY LEGITIMATE DEAL ON ANY AMIGA PRODUCT!

AMIGA PERIPHERALS

1MB FATTER AGNUS CHIP(8372A)	$89
A-501 EXPANSION MODULE	$149
A-1010 DISK DRIVE	$129
A-1011 DISK DRIVE	$169
A-2060 ARCNET ADAPTOR	$219
A-2065 ETHERNET ADAPTOR	$279
A-1084 RGB COLOR MONITOR	**$279**
A-1930 VGA COLOR MONITOR	$439
A-2088-D BRIDGEBOARD	$489
A-2286D AT BRIDGEBOARD	$1079
A-590 HARD DRIVE	$489
A-1950 MUILTISCAN	$549
A-2024 HI RESOLUTION MONO MON. (1008 X 800)	$569
A-2630 ACCELERATOR BOARD	IN STOCK

GREAT VALLEY PRODUCTS

IMPACT A-2000 8/0	$215
28MHz. 68030 ACCELLERATOR FOR A-2000	$769
GVP 3001 KIT (28 MHz.) WITH 68030, 4MB, 68882	$2029
3001 KIT w/QUANTUM 40MB	$2369
3001 KIT W/QUANTUM 80MB	$2649
GVP 3033 KIT (33MHz.) w/68030, 4MB, 68882	$2599
3033 KIT w/QUANTUM 40M	$2939
3033 KIT w/QUANTUM 80MB	$3229
GVP 3050 Kit (50 MHz.) w/68030, 4MB, 68882	$3569
3050 Kit w/Quantum 40MB	$3969
3050 Kit w/Quantum 80MB	$4269

A-2000 HARD CARDS

IMPACT A-2000 HC/45	$499
IMPACT A-2000 HC/40Q.IN STOCK	

ALL OTHER GVP PRODUCTS IN STOCK

AMIGA COMPATIBLE PERIPHERALS

A-MAX MAC Emulator for AMIGA	$109
AMIG-A-TOSH Mac Compatible Drive for A-MAX	$154
AMIG-A-TOSH PLUS	$239
A-MAX ROM	$129
CALIFORNIA ACCESS 3.5" DISK DRIVE	**$126**
COLOR SPLITTER..IN STOCK	
FLICKER FIXER	$429
FRAME GRABBER	$489
FRAME GRABBER SOFTWARE UPGRADE	CALL
MASTER 3-A 3.5" DISK DRIVE	**$115**
MASTER 5-A 5.25" Disk Drive	$189

GENLOCKS

GEN-ONE	$399/PRO-GEN $299
MAGNI-4004/4004S	$1359
MAGNI 4004 or 4004S/4010	$1569
NERIKI IMAGEMASTER PRO	CALL
OMNIGEN 701	$1369
SUPER GEN	$629
SUPER GEN 2000S	CALL

BASEBOARD
Memory Expansion for A-500 (uses A-501 Expansion Slot)

0K	$139	3MB	$379
1MB	$219	4MB	$459
2MB	$299		

MICROBOTICS
Memory Upgrades for A-2000

8up 0K	$159	8up w/6MB	CALL
8up w/2MB	$309	8up w/8MB	CALL
8up w/4MB	$449		

PANASONIC WV-1410 with VARIABLE 16mm LENS wth IRIS	$259
PROGRESSIVE PERIPHERALS 2MB EXPANDER FOR A-2000	**$229**
SHARP JX100 ColorScanner w/Software & Cables	$729

SUPRA RAM 2000

2MB RAM	$259	6MB RAM	CALL
4MB RAM	$399	8MB RAM	CALL

SUPRA 2400 EXTERNAL	$115
SUPRA 2400zi INTERNAL	IN STOCK
SUPRA RAM 500 (512K Expander for A-500)	$77
VIDTECHSCAN LOCK	CALL

PRINTERS

HEWLETT PACKARD

DESKJET	$519.95
DESKJET +	$669.95
LASERJET IIP w/Toner	$989.95
PAINTJET	$949.95
HP LASERJET III w/Toner	$1579

STAR

NX-1000II	CALL
NX-1000 RAINBOW	CALL
XB-2410	$439.95
XB-2415	$569.95
COLOR OPTION KIT for XB PRINTERS	CALL
NEC MULTISYNC IIID MONITOR	**$649**

CITIZEN

GSX-140	$329.95
COLOR OPTION KIT	CALL

PANASONIC

KXP-1180	$174.95
KXP-1191	$214.95
KXP-1124	$289.95
KXP-1624	$429.95
KXP-1695	$419.95

OKIDATA

OKIMATE 20. w/Plug 'n Print	$229.95

CANON

BJ-130E	$579.95

EPSON

LX-810	$199.95
FX-850	$349.95
LQ-510	$339.95

ALL OTHER MODELS IN STOCK! ALL MODELS DISCOUNTED!

HARD DRIVES for AMIGA 500

TRUMP CARD 500 *EXTERNAL CHASSIS ENABLES ANY SCSI HARD DRIVE TO OPERATE WITH AMIGA 500 (EXPANDABLE TO 2MB)....$229*

THESE HARD DRIVE PKGS. ARE PRICED WITH THE TRUMPCARD 500 INCLUDED!

Seagate

ST-125N (20MB)	$429	ST-157N (49MB)	$499
ST-138N (30MB)	$459	ST-157N-1 (49MB, 28MS)	$519
		ST-177N (60MB)	$575
		ST-1096N (80MB)	$669

Quantum

40MB (19ms)	$559
80MB (19ms)	$779
105MB (19ms)	$849

SAME PACKAGE AS ABOVE WITH XETEC FASTTRAK EXTERNAL CHASSIS CAN BE EXPANDED TO 8MB RAM.....ADD $110

SUPRA HARD DRIVES FOR A-500

20MB	$465
40MB	$569
80MB	$839

A-1000 VERSIONS ADD $80
2MB RAM EXPANDER AVAILABLE!

SUPRA HARD DRIVES FOR A-2000

40MB (Quantum)	$529
80MB (Quantum)	$699
105MB (Quantum)	$789

HARD DRIVE CARD PKGS. AVAILABLE..CALL

HARD DRIVES for AMIGA 2000

THESE HARD DRIVE PACKAGES INCLUDE YOUR CHOICE OF XETEC, TRUMPCARD OR SUPRA HARD DRIVE CONTROLLER CARDS

Seagate

ST-125N (20MB)	$349	ST-157N (49MB)	$429
ST 138N (30MB)	$379	ST-157N-1 (49MB, 28MS)	$449
		ST-177N (60MB)	$509
		ST-277N (60MB)	$469
		ST-296N (80MB)	$499
		ST-1096N (80MB)	$599

Quantum

40MB (19ms)	$479
80MB (19ms)	$679
105MB (19ms)	$759

MasterCard | VISA

Certified check, bank check, money orders, approved P.O.'s, Visa, Mastercard, Diner's Club, Am-Ex, Optima, Cart Blanche, C.O.D.'s & wire transfers accepted. Please call before submitting P.O.'s. No additional surcharge for credit card orders. Non-certified checks must wait 4-6 weeks for clearance.. Prices and availability subject to change without notice. Not responsible for typographical errors. Return of defective merchandise must have prior return authorization number, or returns will not be accepted. Please add 5% shipping & handling (min. $6). Orders over $1200 are discounted to 3% shipping & handling Orders over $3000 are discounted to 2%. (Canadian orders please call for shipping rates). APO FPO orders please add 10% shipping & handling. All APO FPO orders are shipped first class priority air. All orders can be shipped Air Express-call for details. D.C.A. #800233. Amiga is a registered trademark of Commodore-Amiga Inc.

Circle 121 on Reader Service card.

Quick response to user requests, achieved through simple yet efficient program logic . . .

Exceptional Conduct

by Mark Cashman

EVERYONE LIKES RESPONSIVE PROGRAMS. BUT HOW CAN A program respond quickly, even when in the middle of complicated operations? The answer is found in a special provision of Exec called exceptions. Exceptions are a task-private temporary redirection of control in response to a signal.

Each task has a set of signal bits in its task control block. There are 32 signal bits contained in a longword of the task control block. This longword, in the Benchmark Modula-2 libraries, is named tcSigRcvd (for task control Signals Received). Sixteen of these signal bits are reserved for the use of Exec.

Each time you call CreatePort to create a message port for your task, a signal bit is allocated for the message port (the number of this bit is stored in MsgPort.mpSigBit, and the corresponding bit is set in tcSigAlloc). Then, when you call WaitPort to wait for a message to arrive at the message port, your task sets the corresponding bit in tcSigWait and is suspended by Exec until a message is received. Then, when the message is received, SendMsg sets the same bit in tcSigRcvd, and Exec, noting that the same bit is set in tcSigWait, schedules the task for reactivation.

It is also possible for a signal bit to cause Exec to invoke a task-specified procedure. This procedure is called the exception procedure. First, it must be specified which procedure is the exception procedure. This is done by putting the address of the procedure in tcExceptCode. Next it must specify which signals will cause the exception routine to be called. This is done by setting the appropriate bits in tcSigExcept. The Modula-2 program below demonstrates this. Note that the MessageUtil, Termination, and Timer modules are my own, and that the other modules are part of the Benchmark Modula-2 libraries.

```
MODULE TestExcept;

  IMPORT

    InOut,
    Intuition,
    Memory,
    MessageUtil,
    Nodes,
    Ports,
    PortsUtil,
    SYSTEM,
    Tasks,
    Termination,
    Timer;

  FROM SYSTEM IMPORT ADDRESS, ADR, BYTE, TSIZE;
  FROM Termination IMPORT Assert;

  TYPE

    ExceptionRoutineTYPE =
      PROCEDURE;

  VAR

    ExceptionRoutine:
      ExceptionRoutineTYPE;

    Iterations:
      CARDINAL;

    KeepRunning:
      BOOLEAN;

    WindowPtr:
      Intuition.WindowPtr;

  PROCEDURE SetKeepRunningToFalse;
    BEGIN

      KeepRunning:= FALSE;
```

```
  END SetKeepRunningToFalse;

PROCEDURE Terminate;
  BEGIN

    IF WindowPtr # NIL
       THEN Intuition.CloseWindow(WindowPtr^); END;

  END Terminate;

          Tasks.SignalSet(CARDINAL(WindowPtr^.UserPort^.mpSigBit));

  END InstallExceptionRoutine;

BEGIN

  Iterations:= 0;
  KeepRunning:= TRUE;

  WindowPtr:= NIL;

  ExceptionRoutine:= SetKeepRunningToFalse;

  Termination.RegisterProcedure(Terminate);

  OpenWindow;
  InstallExceptionRoutine;

  WHILE (Iterations < 100) AND KeepRunning DO
    Timer.Wait(0,0,1,0);
    INC(Iterations);
  END;

  IF KeepRunning THEN
     InOut.WriteString("Iteration termination.");
    InOut.WriteLn;
  ELSE
     InOut.WriteString("Exception termination.");
    InOut.WriteLn;
  END;

  Termination.NormalTermination;

END TestExcept.
```

The program takes a common situation—detecting window close—and eliminates the step of checking the message port each time through the loop by replacing it with the checking of a boolean variable that is set to false by the exception routine.

REAL USES OF EXCEPTIONS

This is a good way to test the use of exceptions, but it is not fully representative of the best way to use exceptions. Here are some real examples:

A fractal generation program responding to a mouse button press in order to zoom in on an area; the exception routine takes control, suspending the calculation for the current pixel, while the user defines the area to be zoomed; then, after starting a new task to display the defined area, the calculation in the current task for the current pixel is resumed.

A file name requester reading a directory to format a display of the files in the directory needs to respond when the user picks another directory from the directory list; a flag is set by the exception routine to restart the directory examination process using the new name.

WHAT MAKES EXCEPTIONS USEFUL?

Basically, these examples—plus the program above—illustrate the three possible uses of an exception routine:

Terminate an iterative process.
Suspend an iterative process for a user action.
Restart an iterative process with new starting conditions.

In all cases, the advantage of the exception routine is the simplicity and efficiency of program logic. In the case of types one and three, what would otherwise be a test of the message port, with the consequent overhead followed by message type dispatching logic, merely becomes the test of a variable. In the case of type two, the existence of the user action logic is invisible to the main loop. In all cases, the loops are simplified, and the checking of the message port only occurs when there really is a message to be read.

•AC•

SNAP, *CRACKLE*, & POP!

FIXING A MONITOR LIGHTNING BUG ON COMMODORE MONITORS 2002, 1902, & 1080

by Richard Landry

[WARNING: THE FOLLOWING hardware fix involves making modifications to a monitor equipped with 20,000 volts. Undertaking such a project can be extremely dangerous, and is therefore recommended for the technically inclined only. Amazing Computing *assumes no responsibility for any damages and/or injuries that may be incurred while performing these modifications.]*

A potential hardware bug that may result in a high-voltage discharge is prevalent in several early models of RGB monitors supplied for C-128 and Amiga computers. Since the advent of the Amiga and C-128 computer, a significant number of users have experienced a problem with a sudden snap, crackle, or popping sound accompanied by a momentary loss of picture on the monitor. Such occurrences are usually infrequent to start with, but become more frequent with the age of the system. Still, in at least one case I know of, an Amiga 2000 system developed the condition almost immediately, and with such intensity that the system would consistently crash.

My Amiga 1000 and the associated monitor were purchased at a Chapter 11 sale, the first year the Amiga appeared on the market. The monitor occasionally cracked an electrical snap and, in late 1988, it gave a snap that evoked the Guru. The snap did not appear for a few months after that. However, in February '89, while my computer system was being used for a demonstration at a club meeting, the electrical snap reappeared. Inquiries did not provide any answers, but the monitor was checked over. During this examination, it was found that taking the cover off caused the snapping to stop. A muffin fan was attached, but about a month later the

computer developed a condition whereupon it would occasionally lock up while attempting to save or print a file. By early April of '89, the frequency of the lockups had increased; the computer could barely be used for an hour before it locked up. A snap was observed after these system lockups occurred.

Intermittent problems are the hardest to correct, but the problem had to be isolated and solved. Frank Gerard of ECS, an authorized Commodore service shop in the Spring Park Lake, Minnesota area, worked on the problem with several different monitors. He concluded that the problem was related to the red high-voltage wire leading from the high-voltage transformer to the anode cap on top of the picture tube. Frank sprayed a plastic insulating coating around the high-voltage transformer where the red wire emanated. This seemed to correct the problem in some of the monitors, but not in all. Bill Hanley, a Minnesota public TV station engineer, brought in some monitors suffering from the same problem. After discussing the problem with Frank, Bill checked several bulletin boards where he located information about a carbon trace being established on the monitor motherboard by arcing.

Frank studied the bottom side of the motherboard and found evidence of carbon traces created by shorts from a heat shield tab to a ground foil on several snapping monitors.

There is a heat sink for a voltage regulator on the side of the monitor near the high-voltage transformer. This heat sink is an "L"- shaped metal plate about four inches high and five inches long which angles at the rear corner for about an inch along the back of the monitor. The heat sink is attached to the motherboard of the monitor by two screws and a metal twist tab that extends through the motherboard. The metal twist tab is very close to the ground foil on the bottom side of the motherboard and, when it is twisted, it comes very close to the ground foil. It doesn't take much voltage to bridge this narrow air gap. This pathway is on the bottom side of the motherboard and out of sight, so it seemed an improbable source for the problem.

We concluded that the large heat sink seems to act as a large capacitor near the high-voltage wire, and when the voltage builds up high enough in the heat sink, it discharges with a small spark to the ground foil. Consistent discharges inside a dirty monitor will help build a trace path to the ground, increasing the frequency and size of the voltage discharge. If the discharge is large enough, the high-voltage on the ground trace will be reflected back through other monitor components and back to the computer. This problem seems to be prevalent on Commodore/Amiga models 2002, 1902, and 1080.

The Amiga 1084 monitor and other RGB Commodore monitors have a smaller heat shield and seem to have a better high-voltage cable path that is farther from possible conducting paths. This seems to prevent the problem from occurring in those monitor models.

Frank Gerard has solved the problem in four steps:

1. Clip off the shorting metal twist tab from the heat sink. The two screws can hold the heat sink adequately. Carefully scrape any carbon traces on the motherboard created by arcing from the tab slot to the ground trace.

2. Provide extra electrical insulation around the red high-voltage wire. Split heat shrink tubing to wrap around the high-voltage wire and use electrical tape to completely encase the wire with extra layers.

3. Spray Koloid Clear Acrylic plastic around the wire and the high-voltage regulator to reduce leaking from high-voltage sources.

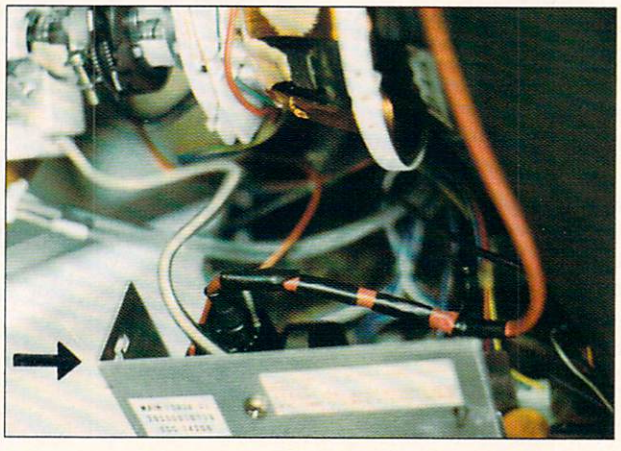
4. Use TV Corona Dope to plastic coat the area of twist tab slot and the ground trace.

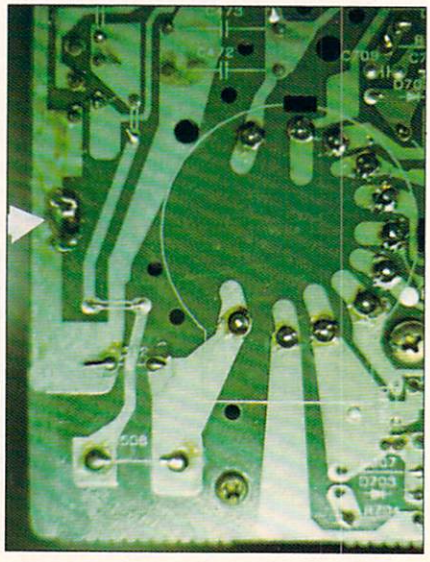
A reasonable charge for having the above performed is about $50.

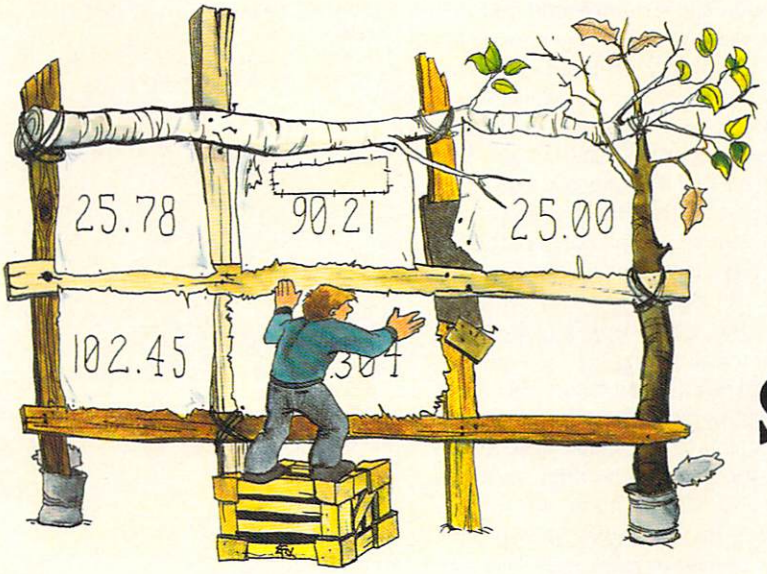
INNER WORKINGS

Figure One shows a view of the top of the L-shaped heat sink (arrow) and the red high-voltage wire. Note the extra insulation on the high-voltage wire in the area of the heat sink. Figure Two shows a view of the left rear corner on the bottom side of the motherboard of a repaired snapping monitor. The arrow points to a ground foil and the slot where the tab from the heat sink comes through the motherboard. This slot has been filled with epoxy. •AC•

Below: Figure One
Right: Figure Two







Poor Man's Spreadsheet

by Gerry L. Penrose

THIS IS SIMPLY AN EXERCISE IN MANIPULATING ARRAYS THAT WAS DEVELOPED FROM doodling around with another idea. Before we get to the description of what is happening here, type in the program. It's quite short. For now, type in the fourth program line with the preceding inverted comma and the result will be a full array READ in from the DATA statements. A query appears at the bottom of the screen; answering yes by pressing "Y" puts up another request. Choose a cell number and type it in, then type RETURN. This will bring a request for an input. Type this in and watch very carefully, as the contents of the affected cells change.

This, basically, is what happens in a full-blown spreadsheet: as you change the content of one cell, the affected cells also change. There is one difference though—in the full-sized job, certain cells have to be filled with formulas before any changes can occur. We are not that sophisticated, and yet this could be the basis for a quite useful program. In keeping with my policy of supplying ideas upon which you can build, I leave the expansion to you.

Now for some explanation of the inner workings. PRINT USING is what locates the contents of each cell. This ensures that decimal points will always show up in the right place and under each other in columns, which makes for a neat and tidy layout. PRINT USING also forces the cell numbers (1–4, etc.) to line up properly, and you will agree that they do look quite neat. It is possible to PRINT the five lines using only one line of code, but the result appears rather ragged. When a program is small like this, there is no need to save bytes (you have plenty of memory available), so indulge yourself. Always keep in mind that there may be times when you need every bit of memory you can grab. These are the times to conserve memory by line crunching, sometimes at the sacrifice of a little aesthetics in the process.

The loop writing the cell numbers uses the counter 'c', adding '3' to each iteration of the loop and so placing our numbers just where we want them. The LOCATE x,y has been mentioned before and is a quite useful tool. Together with counters, such as the one just described, there are many things which can be done in the way of manipulating the positions of various elements of your displays.

The next loop is a case in point. Here we use three counters: the first one, 'y', is used to locate the cells horizontally. The counter 'x' is also used here to set the vertical positioning. Finally, the counter 'pls' adds the elements of the cells and arrives at a total which we can then place in any position we wish, again using the ubiquitous PRINT USING formula.

The next loop is a WHILE WEND loop, which can be used in conjunction with a counter and will replace the FOR NEXT loop. The difference between the two is that you can fall out of a WHILE WEND quite easily on command and without dire results. The FOR NEXT loop is not quite as forgiving. Nine times out of ten you will get away with it, but the tenth time you are likely to find the 'Meditation Guru' waiting for you. The rule would be: a closed loop which must be completed should always be a FOR NEXT. If you want to sneak out of a loop before it has come to its natural conclusion, then use the WHILE WEND.

This loop again uses three counters. This time, the 'pl' counter adds the vertical columns. The 'z' counter does double duty, first in the WHILE WEND loop and again to space the LOCATE elements. The 'y'

counter counts the elements of each column and is returned to zero after each go through the FOR NEXT loop.

The last section, testin:, takes your input and places it into an array element. As you can see from glancing at the program, there is more than one way of filling an array. It can be done, as we have done here, using a number of DATA statements. While not very convenient, this serves well if you have a series of numbers which only need changing or updating occasionally. It means listing your program and adding or subtracting DATA statements as you wish. In producing a program such as this the method was quite useful, in that it enabled me to put in a little demonstration without writing a separate program. To change from the DATA entry to your own entry, simply remove the single inverted comma from the front of the fourth working program line, the one reading 'GOTO enter.

The second way of entering your elements is now open to you. If you run the program with the inverted comma removed, you are faced with an array of cells containing zeroes which you can fill as you wish. As you enter each cell's contents, you will be able to note the changes occurring.

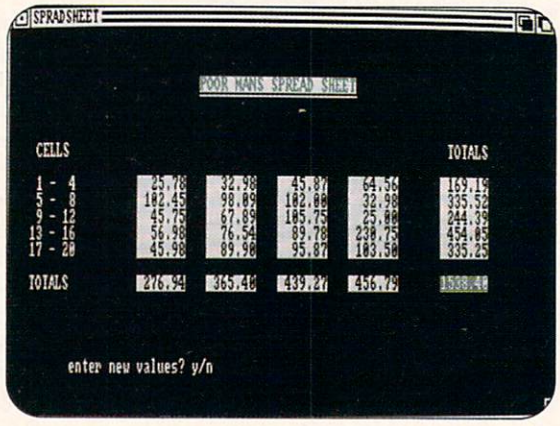
A third method of filling such an array would be to import the complete array from a SEQuential or RELative file. I have covered both of these in my two previous programs, so why not go ahead and devise your own method of taking the contents of an array from this program and placing it in either of the two types of file?

Another suggestion would be to increase the size of the array. I purposely left spaces between array elements, so that you could see how it was put together. It doesn't have to be this way you could have the elements abutting each other. You could also enlarge the sizes of them—they are controlled by the PRINT USING command—and the spacing of elements by use of counter 'y' in the LOCATE sections. Experiment, but make sure you have saved your original working copy so that you can get it back if you goof.

A final word about color. The use of color in a program serves to accentuate certain elements of your display and makes for a more readable display than one that is only blue and white. I think you will see what I mean by the display shown here. There is no need to be mediocre—use your imagination and have fun.

Listing

```
'Poor man's Spread sheet
'by - G.L.Penrose
'Oakville,  Ontario
'Canada
'for AMAZING COMPUTING
DIM p(25) ,pls(25),tl(25)
,pl(25),t(25)

start:
a$="####.##":z$="##"
'GOTO enter
DATA  25.78,32.98,45.87,64.56,102.45
DATA  98.09,102.00,32.98,45.75,67.89
DATA  105.75,25.00,56.98,76.54,89.78
DATA  230.75,45.98,89.90,95.87,103.50
FOR i=1 TO 20 :READ p(i) :NEXT i

enter:

z=0:x=0:y=0:c=0:
v$="POOR MANS SPREAD SHEET"
LOCATE 8,5
PRINT  "CELLS"
LOCATE  4,40-LEN(v$)/2
COLOR 3,1
PRINT  v$
COLOR 1,0
LINE  (224,33)-(400,33)
FOR i=1 TO 5
LOCATE 9+i,4
PRINT USING z$;(i+c)
LOCATE 9+i,6
PRINT " - "
LOCATE 9+i,9
PRINT USING z$;(i+3+c)
c=c+3
NEXT i
LOCATE 8,64
PRINT "TOTALS"
LOCATE 16,4
PRINT "TOTALS"
COLOR 2,1
 FOR i=1 TO 5
   FOR j=1+y TO 4+y
      LOCATE 10+x,10+(j-y)*10
      pls=pls+p(j)
      PRINT USING a$;p(j);
   NEXT j
  PRINT SPC(6) USING a$;pls
  y=y+4
  x=x+1
  pls=0
 NEXT i
y=0
z=1

WHILE z=<4
    FOR i=z+y TO 20 STEP 4
       pl=pl+p(i)
       y=y+4
    NEXT i
  LOCATE 16,10+(z)*10
  PRINT  USING a$;pl
  pt=pt+pl
  pl=0
  y=0
  z=z+1
  LOCATE 16,63
  COLOR 1,3
  PRINT USING a$;pt
  COLOR 2,1
WEND

pt=0
COLOR 1,0
LOCATE 21,10
PRINT "enter new values? y/n"

ask:
q$=INKEY$ :IF UCASE$(q$)="" THEN ask
IF UCASE$(q$)="N" THEN CLS :STOP
IF UCASE$(q$)="Y" THEN contin
GOTO ask

contin:
LOCATE 21,10
PRINT  SPACE$(35)

testin:

LOCATE 19,10
INPUT "Choose cell. "; x
LOCATE 20,10
INPUT "Amount ." ;a
p(x)=a
LOCATE 19,1
PRINT  SPACE$(35)
LOCATE 20,1
PRINT  SPACE$(35)
GOTO  enter
```



Looks great—and easy to read, too!

•AC•

upgrades • fixes • updates • new releases

bug bytes

by John Steiner

ACCORDING TO A NOTE ADdressed to me via EMail on People Link, the new Perfect Sound version 3.0 from Sunrize Industries has a couple of problems. First, it does not work correctly with 68020 or 68030 accelerator boards. It cannot take keyboard input correctly when running under 68020 or 68030 mode. This causes the gain controls in the software to not work right and makes other functions operate sporadically. The program works fine in 68000 mode.

Anthony Woods of Sunrize has said they are aware of the problem and are working on it. Also, AudioMaster II from Oxxi/Aegis Development does not support the latest version Perfect Sound. A call to Oxxi confirmed that fact, and they commented that they are currently in development of AudioMaster III,which does work just fine with Perfect Sound 3.0. It should be ready sometime in July according to a spokesperson.

She commented that the upgrade fee for currently registered AudioMaster II owners would probably be about $25 or $30 Contact Oxxi directly about the upgrade. As far as a Perfect Sound 3.0 software upgrade to run on the 68020 and 68030, Mr. Woods advised the writer to check back with him in about a month as it should be ready by then. He expected that there would be no charge for registered users to receive the upgrade/bug fix. *Sunrize Industries, Box 1453, College Station, TX 77841, (409) 846-1311. Inquiry #200.*

SCOTT BUSSE WROTE AN EMAIL letter regarding the problems I had reported with incompatible ANIM formats in an earlier Bug Bytes. It seems that minor differences in the way animation generating programs store ANIM files are causing problems when graphic artists try to load ANIMs into different applications from which they were created. Scott offered a solution to the problems in a shareware-program. The Animation Bridge is a $20.00 shareware program that currently supports Anim-5 files created from:

Photon Paint2.0	The Director
Videoscape 3D	Movie2.0
Animation: Editor(v1.11)	Cel Animator
DeluxePaint III	Turbo Silver 3.x
AniMagic	Page Render 3D
Animation Station	Sculpt-Animate 4D

According to Scott, the program will save the files from the above programs in a format that can be read into:

DeluxePaint III	The Director
AniMagic	Animation: Editor (v1.11)
Animation Station	Movie 2.0

The program will also play ANIM files of any variation within the Anim-5 spec. If the animation program you are using isn't on the above list, chances are it doesn't need fixing. Contact Scott via CompuServe ID 73040,2114 for more information.

SINCE THE ANNOUNCEMENT AND pre-release of the Amiga 3000, I have been getting letters and electronic mail reporting problems with programs that don't work under Workbench 2.0, or under the specially modified Workbench 1.3.2 that is currently only available on these early production dealer demonstration units.

The A3000 systems that dealers currently have do not contain operating system ROMs like the Amiga 2000 and 500 series. Workbench (either 2.0 or 1.3.2, if desired) is loaded into RAM from the hard disk at present as the version 2.0 operating system provided with these units is still in pre-release state. Further, upgrades have been promised by Commodore to be shipped on a monthly basis to the dealerships that have demonstration Amiga 3000 systems.

When Workbench 2.0 is finalized, it will be committed to ROM and all of the pre-release version 3000s that dealers have will be upgraded to full production unit status. At the Amiga 3000 introduction I attended in Chicago, a Commodore representative commented that there could still be deletions and additions to the features that are being included in Workbench 2.0. Even the Workbench 1.3.2 that has been patched into the 3000 is also somewhat non-standard, and will not be part of the A3000 when 2.0 is finally released so there may be problems with running current 1.3 applications that work fine on other systems.

As a result of the currently unfinished nature of the A3000, I have decided that it serves no useful purpose to point out problems with currently-available software on the beta version 2.0 operating system, or on the Workbench 1.3.2 that won't be available on the production units. I would expect that many of these problems will disappear when 2.0 matures to final release state. It would be unfair to the developers of existing software to point out problems with their software that may not even be there when end users are finally able to purchase the A3000.

The confusion and possible lost sales would only be a detriment to the Amiga development community, especially when the problems may be fixed before the computer is available for sale to the general

public sometime in July. Besides, in my own testing of the A3000 under Workbench 2.0, it might take less space to list those programs that work 100% than to list those that don't. A lot of software has problems at this point, which is probably why Commodore went to the work of patching 1.3.2 into the A3000. It allows dealers to be able to demonstrate working applications on the computer for prospective customers.

Once version 2.0 is available in ROM, companies with software that still have problems under 2.0 will have to deliver upgrades. I am expecting that a lot of programs will need minor "tweaking" to make them work 100% under the new system. Expect a flurry of upgrade notices here in Bug Bytes over the next few months as developers complete these upgrades. As of this writing, Workbench version 2.0 upgrades for existing 2000 series computers won't be available until September according to a Commodore press release.

There seems to be some confusion about being able to upgrade A500 systems to the 2.0 Workbench, as Commodore didn't specifically mention them in the press release. Other material I received from Commodore stated that the A500 upgrades would be available "at a later date." Some people have become upset over a concern that since the A500 is going into the mass market that Commodore didn't plan to upgrade the units to run 2.0.

There is reason that Workbench 2.0 features cannot be made available for the 500, and there is no benefit for Commodore to refuse to upgrade the Amiga 500 systems currently in service. Version 2.0 will be worth the wait, let's give them time to get it working right before it's released.

OXXI, INC. HAS ANNOUNCED THE release of version 1.5 of VideoTitler version 1.5. According to a press release dated April 20, 1990, the new version includes 3D text manipulation, built-in animation facilities and a re-designed user interface. The program now allows text to be stretched into various 3-D perspectives with the use of "handles."

An "Extrude" function works with the program's Poly-Fonts to allow text to take on the properties of objects which can be stretched, spun, rotated or distorted. Animations allow text to spin, circle and change colors in any direction and length. Key frames are generated that allow the user to specify the start and end points, and the program will generate all of the in-

between frames. An improved user interface makes animation generation much more easy to accomplish and a new "tool box" gives the user instant access to a variety of commands without using pull-downs.

Additionally, the program now comes with Lights! Camera! Action!, a presentation generation utility at no extra charge. The program should be shipping by the time you read this at a retail price of $159.95. Registered users of the current version 1.1 or earlier may upgrade directly from OXXI for $34.95. Lights! Camera! Action! can be included with the upgrade, the total price is $39.95. While I was confirming information about Oxxi's upgrades this month, I was asked by their technical support person to make a couple of comments regarding MaxiPlan. Intuitive Technologies is the company that markets MaxiPlan III. I have commented in previous Bug Bytes columns that people have had trouble obtaining adequate technical support from Intuitive Technologies. Oxxi, Inc. is currently marketing both Maxi Plan 500 and Maxi Plan +, which were both written by the same person who wrote MaxiPlan III. Oxxi has been getting lots of calls wanting to know why they aren't supporting their products, or demanding to know the status of their upgrade orders. Please note that Oxxi is currently supporting their products 100% and that they want our readers to know that Intuitive Technologies is not the same company as Oxxi.

If you are a registered owner of either MaxiPlan 500 or MaxiPlan +, (currently in version1.9) you may obtain technical support directly from Oxxi at the number listed below. I wish to apologize to Oxxi for any confusion this has created for them. I was not aware that any other company was marketing MaxiPlan in any other format than MaxiPlan III from Intuitive Technologies. *Oxxi, Inc. Box 90309 Long Beach, CA 90809-0309, (213) 427-1227, FAX (213) 427-0971. Inquiry #201.*

That's all for this month. If you have any workarounds or bugs to report, or if you know of any upgrades to commercial software, you may notify me by writing to:

John Steiner
c/o Amazing Computing
Box 869
Fall River, MA 02722 ...
or leave EMail to Publisher on People Link or 73075,1735 on CompuServe

•AC•

PD Serendipity

Insight into the World of Public Domain Software for the Amiga®

THIS MONTH I WOULD LIKE TO TAKE A brief look at some editors: IE, an icon editor; AZ, a text editor; and MED, a music editor.

AZ V 1.50—an update to version 1.40 on Fred Fish disk #228—appears on Fred Fish disk #346. The upgrade includes some new features in addition to bug fixes.

AZ is a text editor, meaning the files it produces can only contain characters. The files created can be assembled, interpreted or compiled as they are. All the characters in the active keymap (up to 256) can be entered.

AZ allows you to open as many Windows as needed. The number of Windows allowed is limited only by the memory available. Multitasking is supported. You can start a job in one window, let that run, then start working in another.

There are two versions of AZ, the big version and the short version. The difference is in the big version, the FileRequester code has been included in the code at link time; therefore, no installation—just click on AZ's icon. To install the short version, simply copy the "isup.library" into your LIBS: directory.

AZ utilizes the function keys and supports simple quick key commands, such as <Amiga> <Q> to quit. It also allows the use of the <option> key plus another key to toggle between colors, length of a line, etc.

I found this program and the manual easy to use. The manual includes a list of all changes made through the updates, and explains the program very well.

Author:Jean-Michel Forgeas

The next program is an icon editor, **IE** V1.0 (FFD 342). IE creates icons up to 640X200 pixels, handles creating and editing of dual-rendered icons, and allows you to preview the icon before ending the program.

I found this program lots of fun to experiment with. You draw with an icon in the shape of a paint brush, select from one of the four different colors available, and create your personalized icon.

Some features include "Flood Fill", which fills in the selected area with the chosen color, and iconify. Another feature lets you copy the image and then choose one of the rendering options, "non-select" to select, or "select" to non-select. This makes your icon appear to move when clicked on. Use preview mode to test your icon.

A new version of IE is in the works. Possible features will let you read IFF pictures as icon bitmaps and write icons as IFF pictures; also, a better file requester will be added. *Author: Peter Kiem*

MED V2.00 (FFD 349), an update from V1.12 on FFD 255, is a music editor which helps you create songs. Some features and bug fixes include a new player routine made with Assembler, fast screen rendering routines, new user interface with file requester, MIDI support and up to 16 tracks, insert/delete blocks, extended keyboard, and song packing, to name a few.

MED is similar to the public domain program SoundTracker and supports some of SoundTracker's features, and more. MED is written in C (Lattice C V5.04), except for the most important routines done in Assembler. MED supports multitasking and you can play songs created with it on MEDPlayer.

Author: Teijo Kinnunen

by Aimée B. Abren

UPDATES

SOFTFONTS

Softfonts (FFD 342) is an update from FFD 327. It converts portrait soft fonts for HPLaser-jet compatible laser printers to landscape format.

Softfonts works through Intuition or through the CLI. The major update corrects a fault with bitmap rotation. Includes source.
Author: Thomas Lynch

CROBOTS

CRobots V2.3w (FFD 345) is an update to V2.2w on FFD 331. CRobots is a game based on computer programming. Gameplay involves designing and writing programs (in C) to control a robot whose mission is to find and destroy other robots. All robots are equally equipped and up to four robots may compete at once.

Changes include added checks for files being write/read/delete protected, improved IF-THEN generating section of the compiler, and new flag "NOWAIT" to turn off the "press any key to continue" messages. This allows complete operation from within a script.

CRobots consists of a C compiler, a virtual computer, and battlefield display. Requirements are 512K, DOS 1.3, ARP1.3 and a text editor. Includes binary only.
Author: David Wright

GET IMAGE

Get Image (FFD 345) is an enhanced version from FFD 14. This utility program converts DPaint brushed into C source code as image structures.

Changes include setup of the Plane Pick value in the Image structure, and deletion of any unused bitplanes to save memory and disk space. Includes source.
Author: Mike Farren, enhancements by Chuck Brand.

MEMFRAG

MemFrag (FFD 345) is an update from FFD 69. This program displays number of memory chunks/sizes to show memory fragmentation. Chunks are displayed as 2**N bytes. Includes source.
Author: Mike Meyer, enhancements by Gary Duncan

UNSHAR

UNSHAR V1.3 (FFD345) is an update from FFD 287. Unshar is a utility that extracts files from UNIX share archives.

Some changes include a bug fix in "Overwrite (Yes, [No!], Alt)" prompt and a bug fix in code to skip existing files. Also, Unshar no longer exits immediately on file read error. Includes C source.
Author: Eddy Caroll

TEXT PAINT

Text Paint (FFD346) is the second major release of the ANSI editor. Enhancements include the possibility to reload ANSI files or CLI modules, 4 color option, optimized keyboard layout, new drawing modes, right mouse button support, etc.
Binary only. *Author: Oliver Wagner*

CES: Chicago '90

Commodore Dynamic Total Vision & More!

by Andy Patrizio

THE SUMMER CONSUMER ELEC-tronics Show (held this year on June 2–4 in Chicago) annually features products that represent at least the early fruition of stunning new breakthroughs in technology. It is an exposition dedicated to introducing these products to both industry buyers and members of the press. Commodore's presence at CES this year profoundly underscored their commitment to providing the most innovative of computer technologies to the consumer markets.

It was here, in small enclosed area within their larger display, that CBM introduced the Commodore Interactive Graphics Player, a small device already described by some as being perhaps the next great intellectual appliance.

Commodore has created a single new technology from two existing ones—laser disc technology, and the multimedia capabilities of the Amiga. This new technology—Commodore Dynamic Total Vision—is incorporated in the new Commodore Interactive Graphics Player.

One thing should be stressed about the CDTV player: it is **not** considered a computer, nor is it a compact disc player. Although it has the internals of an Amiga computer, and is capable of playing compact discs, CDTV represents not so much a step forward in technology as a sidestep onto a brand new path.

In CDTV, Commodore has created an entirely new market by joining CD technology with Amiga power in a component featuring a simplified interface. The interface is designed not to scare off the otherwise computer-phobic user. In fact, the simplified interface centers on perhaps the most familiar home electronic device in use today—the remote control!

No monitor, keyboard or mouse is needed. The CDTV player connects to a television, and with the remote control, it can operate and access a 550-megabyte CD-ROM drive. A variety of CD reference libraries are already under development, including the Bible, a cookbook, and an encyclopedia. As you might imagine, with CDTV these resources can now include sound and animation.

THE BIRTH OF CDTV

CDTV was officially unveiled at C.E.S. on June 4 by CBM Chairman Harry Copperman, along with Irving Gould, Chairman of the Board, Commodore International, and Nolan Bushnell, whose name is synonymous with computers.

Now the General Manager of Commodoré's new Consumer Interactive Products division, Mr. Bushnell created the first video game ever—"Pong"— back in 1972. He went on to found the Atari Corporation and the Chuck E. Cheese "pizza theater" restaurants for children.

Declaring his high aspirations for the machine, Mr. Bushnell said "It's going to be in the home, it's going to be in the school, it's going to be in industry, it's going to be everywhere."

The technology actually makes the machine "a 21st century library," he noted. Now, a person will be able to see and hear a digitized audio/visual recording of Martin Luther King's "I Had a Dream" speech, instead of only being able to read it from a page.

CDTV will be released in September, with a selling price of $899—rather expensive for a CD player. But as Bushnell points out, "We now have, for the first time, an Amiga platform wrapped around a compact disc."

Mr. Copperman also announced CBM's plan to put a separate sales force in place specifically geared toward schools, to quicken the move of CDTV into the educational market, as well as to take orders directly from schools in an attempt to increase Commodore's share of that vital marketplace.

Designed to look like a VCR, the CDTV player will fit right in with a stereo, receiver, amplifier and television as one component of a total entertainment system. When not functioning as an Amiga, it can play all musical compact discs, with 8X oversampling. The player operates as a computer internally, but it is used like a stereo component externally. Now, anyone has computer capabilities.

IT'S AN AMIGA, TOO

For users who do want standard computer access, add-on peripherals make this an Amiga 500 with one meg of RAM. Peripherals include a keyboard, mouse and drive and the infrared bus. A trac ball is also available. Each peripheral will sell for under $50 except the external floppy drive, which will sell for $199.

All of the peripherals (except the drives) are wireless, and communicate via infrared signals. To permit operation of several devices on the CDTV player, the mouse, joystick and keyboard all send their signals through the "brick," since multiple infrared devices can easily have their signals crossed.

As for a monitor, the CDTV player comes with RGB output, PAL output and NTSC output; it was very inexpensive to offer all options. The player also has a few added features. One is a MIDI in/out port that can be factory installed. Another feature is a DMA slot, for hard drives or SCSI controller cards.

Some compromises have been made—as of press time, you will not be able to play your favorite Led Zeppelin CD and utilize the external floppy disk drive at the same time. And there is no 86-pin bus as on the regular A500, so expansion is limited. But the machine does have open architecutre for future expansion. Once the industry establishes a standard for full motion video, you will be able to adapt your CDTV player.

The benefits of using CD technology are obvious. Software companies no longer have to worry about their products being pirated. Users don't have to deal with VirusX or DiskDoctor to fix infections or bad sectors. Program developers get lots of extra space to design incredibly larger, more complex, more interesting and useful games and applications, with 550 meg.

While CDTV is an attempt to create or define an entirely new market, the player is still an Amiga, and Amiga users will be able to utilize the CDTV software on their present machines. Because the CDTV player has a 68000 chip and one meg RAM, it was not developed for high-level video work, although software may be developed for that purpose.

Therefore, Commodore will release a CD-ROM peripheral shortly after the CDTV player to use programs made for CDTV on any Amiga, save the A1000. The CD-ROM—along with an A2500 or A3000—probably remains a better choice for those who want CD-ROM software for high end multimedia/video use, since the player as a computer is very limited.

The CDTV player was developed to be as little like a computer as possible, both in terms of its appearance and
its operation (that is why it was unveiled at C.E.S. and not Comdex). This is not for the power user or the cable television station.

NOT FOR THE COMPUTER MARKET

Because the player is not being marketed as a computer, don't look to your local dealer. The target retailers will be high-end audio/video retail dealers, department stores and selected retail chains, although just who these are has yet to be specified.

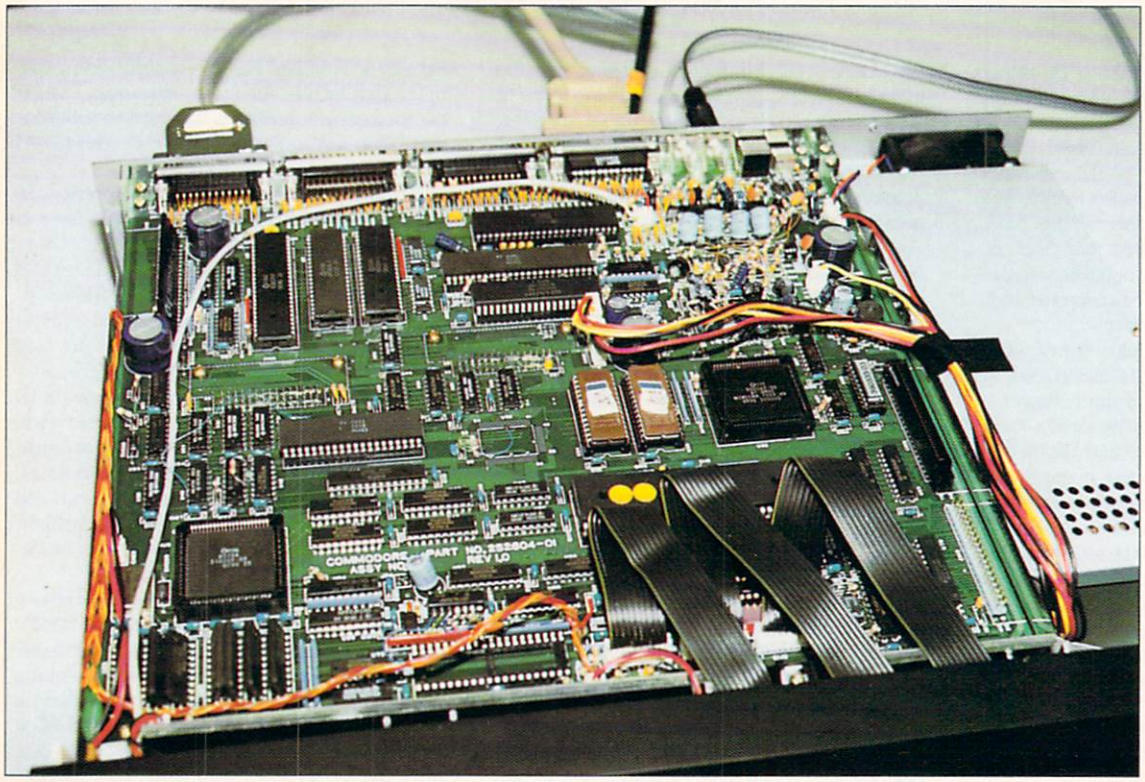
Commodore recently started the Commodore Express service for the Amiga 500, and the CDTV player will also be supported by this service. A 24-hour, toll-free "800" line is available for any problems a purchaser may encounter, and CBM has teamed with Federal Express to provide door-to-door pick-up and delivery for any repairs covered as part of the new service.

CDTV's target market is a new one specifically created with the birth of this new machine. With the interactive reference databases now under development, the Commodore Interactive Graphics Player will look and perform like something from the set of "Star Trek."

For example, one program will be based around the Silver Packette Cookbook. If you plan a meal for six guests, and nine show up, the player will recalculate the proportions needed for that number of guests. If you are out of a required seasoning, it will suggest alternates. And for European users, the machine converts U.S. measurements into Metric.

CBM's idea is to bring computer power to that portion of the consumer market that has so far shied away from computers. Considering Amiga's audio/visual strengths and the fact that operating CDTV is as easy as changing channels on a television with a remote, this idea appears to be a winner. Having taken the look and normal operational procedures of most computers out of this machine, CBM may very well succeed in attracting buyers who normally would not touch a computer.

Above: Commodore Interactive Graphics Player motherboard.
Below: CDTV player with optional peripherals.



Commodore estimates over one hundred titles will be available by release time in September, and double that number by Christmas. The discs are expected to sell for around $30 to $100, depending on the program.

Commodore
Business Machines
1200 Wilson Drive
West Chester,
PA 19380
(215) 431-9100
Inquiry # 336

Amiga Entertainment Software at CES

Games and entertainment software appeared to be the main computer application available. Here is a few of the CES attendees and their latest entries into the Amiga market

Accolade

Jack Nicklaus' Unlimited Golf & Course Design is described as the complete golf experience, containing all the realistic elements of championship play as well as all the tools needed to design challenging and visually stunning holes and courses. Included is *The Bears Track,* an oceanfront 18-hole course designed by Jack

Nicklaus and his team exclusively for this product, and a re-creation of *Muirfield Village,* a Jack Nicklaus designed course and site of The Memorial Tournament. Due in September. $59.95. **Inquiry # 271**

ISHIDO:The Way of Stones provides one or more strategists with the challenge of accurately placing a pouch of 72 stones on a 96 square game board. Each stone is decorated with a symbol and a color.and can only be placed next to another stone that matches either it's symbol or color. As the board fills with stones, the game becomes more complex as players are faced with matching two, three, and even four sides of some stones. The most desirous move, and the most difficult, is a "four-way" where one stone is matched on all sides by four others. *Best Strategy Game of 1989.* Due in July. $49.95 **Inquiry # 272**

Accolade
550 South Winchester Blvd.
San Jose, CA 95128
(408) 985-1700
FAX (408) 246-0885

Capstone

Tom Clancy fans will be psyched to know their favorite author has another title coming out as a computer game. ***The Cardinal of the Kremlin***, due this September. You must find a missing SDI scientist and protect "The Cardinal," America's most secret spy in Russia. *Glasnost,* anyone? $49.95. **Inquiry # 273**

Bill and Ted's Excellent Adventure will feature digitized films and sounds from the actual movie, and is scheduled to be released around the same time as the sequel this July. EXCELLENT! $39.95 **Inquiry # 274**

Another arcade translation to computer screen is ***Superman.*** If it looks as good as the arcade version, this will be a great one- or two-player game. $39.95. **Inquiry # 275**

For all you gamblers, ***Trump Castle*** will provide the ultimate in casino simulations. The game will feature blackjack, roulette, craps, keno, and nine different slot machines. $39.95. **Inquiry # 276**

Capstone A division of
IntraCorp,Inc.
14160 S.W. 139th Court
Miami, FL 33186
(305) 252-9040
(800) INT-RACO
FAX (305) 255-1205

Data East USA, Inc.

The Dream Team: 3 On 3 Challenge. All-pro basketball players Patrick Ewing, Dominique Wilkins, and James Worthy in a fast paced three-on-three style basketball game. Direct access to instant team and individual player stats, through a direct modem to USA TODAY'S Sports Center to keep game play as close and as exciting to the real thing as possible. To be released in late fall. **Inquiry # 277**

Bo Does Baseball provides instant access to real statistics just as the other MVP Sports simulations games. This feature offers computer games with a unique, more true-to-life sports experience. August. $39.95. **Inquiry # 278**

ABC's Monday Night Football (Version 1.5). The complete line up of Data East MVP Sports provides an "in the game" play perspective. With improved broadcast-style statistics, players can keep track of the scoring drive, including downs, yardage stats, looses, and pass completions for more accurate game play. August. $49.95. **Inquiry # 279**

Super Hang On is a fast-paced motorcycle racing game that takes players on a grueling motorcycle circuit that spans the globe. Game points are awarded for overall precision driving in distance, speed and course completion. Players receive bonus points for finishing a "perfect run." $44.95. **Inquiry # 280**

North & South™ is a new strategy game with four stages of battle, players control different numbers of armies and territories, launch attacks and travel from state to state as they confront and conquer battle challenges in an effort to win the war. $44.95. **Inquiry # 281**

(continued on page 74)

COMMODORE INTERACTIVE GRAPHICS PLAYER

Technical Specifications

CPU Motorola 68000
CPU Speed 7.15909 MHz (NTSC)
7.09379 MHz (PAL)
Memory 1 megabyte chip RAM
2K non-volatile RAM (reserved for system—clock, prefs, etc.)
512K ROM
Internal Slots intelligent video slot w/15 pin edge connector (for optional genlock, RF board, etc.)
Video Outputs
- digital RGB,analog RGB (DB-23 connector)
- composite video NTSC or PAL (RCA-type connector)
- component video Y-C (S connector type for S-VHS and Hi8)
- RF modulated (F connector)
- optionalgenlock capabilities via plug-in module; three-mode (CD, video source or mixed) under software control

CD ROM Drive Specifications
Sony/Phillips type CD-ROM standard mode 1, mode 2
Data readout from disc
- 153 KBytes/sec (mode 1)
- 171 KBytes/sec (mode 2)
- 2 Megabytes/sec (burst)

Average access time 0.5 seconds
Maximum access time 0.8 seconds
Standard supported ISO-9660
Data capacity approx. 550 MB (equivalent to about 700 Amiga floppy disks)

CD Audio Specifications
8x oversampling
Frequency response 20–20KHz
Maximum audio capacity about 14 hours—AM quality
Sample rate variable from CD audio rate (44KHz) to 6KHz
Dual 16-bit D/A converter plus 10-bit of attenuation

Rear Ports
- Centronics Parallel interface
- RS-232 Serial interface
- External floppy disk drive interface (Amiga floppy disk drive compatible)
- Hardwired alternative to IR for keyboard, mouse, joystick
- 2 audio output ports (RCA-type plug); requires external audio amplifier
- MIDI In/Out

Front Port
- Stereo headphone jack
- Port for optional personal RAM card (up to 64K)

Front Panel Controls
- Power On/Off
- Headphone volume Up/Down
- Play/Pause
- Stop
- Forward/Reverse and Scan/Skip
- CD/TV
- Reset

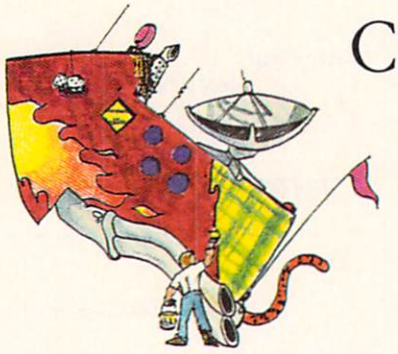
Operating System
- Amiga Kickstart 1.3 in ROM
- ISO 9660 File System Handler
- High-speed decompression for graphics, audio and other data

Infrared Remote Unit Specifications
10 function keys plus Shift key (20 total)
Up, Down, Left, Right movement buttons
Two select keys
CD Audio Play/Pause, Forward, Reverse, Headphone Volume and Stop keys
Computer reset function

Optional Accessories
- External floppy disk drive
- Trackball (Infrared)
- Joystick (Infrared)
- MIDI In/Out, through (third party)
- Personal RAM or ROM card
- Genlock
- Expansion module to house hard disk drive, modem, floppy disk drive
- Keyboard IR (infrared) interface
- Keyboard
- Two-player IR interface
- Modem
- Printer

Getting To The Point:

Custom Intuition Pointers in AmigaBASIC



by Robert D'Asto

AFTER PROGRAMMING WITH AMIGABASIC FOR A WHILE, IT BECOMES APPARENT THAT THERE IS NO direct method for including a customized mouse pointer within application programs. Of course, it is possible to alter the pointer with Preferences, but that only changes its appearance on your machine. When the program runs on a different Amiga, the pointer will appear as it has been set up on that particular machine. It's my guess that most users stick with the default pointer provided by Intuition or something very similar.

It's not that there's anything wrong with a little red arrow. It's actually a very good one-size-fits-all sort of compromise, but there are situations where other types of objects would be more suitable or just more attractive. Changing the form of the Intuition pointer to suit a particular application can significantly enhance the program's utility, ease of use or simply because it looks nice.

If, for example, a paint program was being written, wouldn't it be nice if the pointer were rendered as a pen or paint brush? Or, how about a cross hair type object for those arcade shoot'em ups? Or, maybe a pointing hand or flashlight for making menu selections? The possible variations are endless. Look at the good effects customized pointers make in commercial and PD software.

Customizing the pointer with Amiga-BASIC is actually pretty easy. All it takes is a few library routines and a little knowledge of the makeup of your pointer.

The mouse pointer is a sprite, created and maintained by the section of the operating system known as Intuition. A sprite is a special kind of animated Amiga object which can be further divided into two types: hardware sprites and virtual sprites. It isn't necessary to get into all the technical details of these objects here, so just be aware that a sprite (either kind) is a graphic object which can be animated and must adhere to certain physical restrictions. It can be no more than 16 pixels wide, though it can be as high as the entire display screen. Sprites are also limited to four colors, one of these always being transparent. (ie. the same color has your background)

The difference between the two types is that only eight hardware sprites, numbered 0 to 7, can be defined at any one time. Virtual sprites have limitations as well, but they not pertinent to the scope of this article. Anyone who has ever made sprites with the ObjEdit program on the Extras disk has seen virtual sprites. The Intuition pointer is a hardware sprite, specifically it is hardware sprite 0.

Each of the eight hardware sprites derives its color from a specific set of four color registers or

palette numbers. The Amiga operating system assigns palette colors 16 through 19 to hardware sprite 0 (Intuition pointer). Register 16 is always transparent and numbers 17 through 19 are visible colors. This provides the first and simplest method of customizing the pointer with AmigaBASIC source code. Its color can be changed with PALETTE statements, specifying color registers 17 through 19, to any colors we wish. This works even with a screen that has only 2, 4, 8 or 16 colors. It's still that same old arrow, but at least this gives it a new coat of paint!

Changing the shape of the pointer requires a bit more programming and some help from system library routines. There are four routines in all: Two from the intuition library and two exec library functions. It is assumed that the reader is at least somewhat familiar with the LIBRARY statement, bmap files and calling Rom Kernel routines. If not, this information can be found in the AmigaBASIC manual and in the About Bmaps program on the Extras disk. The listing provided at the end of the article assumes that both "intuition.bmap" and "exec.bmap" files exist in either the :Libs directory of the Workbench disk or in the same directory as the listing source code.

The key routine for creating a custom pointer is called SetPointer and resides in the intuition library. This routine requires six parameters and its calling syntax is:

```
SetPointer& W&,Pointer&,height,width,XOFFSET,YOFFSET
```

Looking at each of these parameters in greater detail, the W& variable is the address of what's known as a Window Structure. This is a list of data residing in RAM which contains a complete description of a window. The AmigaBASIC function WINDOW(7) provides this address so you can simply plug "WINDOW(7)" into this spot when using the SetPointer routine.

The Pointer& value is a long integer which holds the address of a special list of data residing in RAM that describes the shape and colors of the pointer. This list is known as a SpriteImage Structure and is described in more detail below.

The height and width parameters are the overall dimensions of the pointer measured in pixels. The last two values, XOffset and YOffset, give the relative position of the pointer's "hot spot". This is the specific point that the system recognizes the pointer to be.

Taking a closer look at the Pointer& parameter mentioned above, this long integer variable holds the memory address of a special list of data in RAM. This list, called a SpriteImage Structure, is a sequence of two-byte integers (words) in RAM which provides the necessary pixel information for describing the shape and colors of a sprite. It is as long as it needs to be to include all the data of the particular sprite rendered. The first two words and the last two words are always set to zero when this structure is used for describing pointers. The actual image data begins at the third word. A very simple example of a SpriteImage Structure would be:

```
0000000000000000,0000000000000000
1111111111111111,1111111111111111
0000000000000000,0000000000000000
```

For purposes of clarity, the words have been illustrated in pairs. This makes it easier to work with the image data, as will be more apparent in a moment. In actuality the words are simply arranged in sequence from first to last in RAM.

A word is 2 bytes or 16 bits. This structure contains six words. As mentioned above the first two and last two words are always set to zero. The pointer's image data in this example is contained in the third and fourth words. What has been presented is a very simple pointer, one pixel high and 16 pixels wide.

The image data words work like this: The leftmost pixel of the pointer is represented by the leftmost bit of both the third and fourth words. These two bits are both 1, so this pixel is represented by the binary number "11" which equals 3 in decimal. The operating system then assigns a color to this pixel according to this table:

binary	decimal	color
00	0	16 (transparent)
01	1	17
10	2	18
11	3	19

In the above example all the bit combinations are "11" (decimal 3) so the pointer image represented is simply a horizontal line 16 pixels wide (the maximum width of a sprite) rendered in the color assigned to register 19. If the color needed to be changed to register 18, the third and fourth words in the above example would be changed to:

```
0000000000000000,1111111111111111
```

which makes each bit combination "10" (decimal 2), so register 18 would be used for each pixel. Yes, each bit combination in this example would be "10", not "01". It works like this: First, take the leftmost bit of the word on the left (a "0") and then take the leftmost bit of the word on the right (a "1") and mentally place this second bit to the LEFT of the first bit, which results "10". It may seem to be easier and more natural to do it the other way around, but that's how these bit patterns work.

This is why the words of the SpriteImage Structure are shown in pairs. Each horizontal line of pixels which makes up the entire image of the pointer requires two words of image data. This provides each pixel of the image with two bits to define its individual color. Each pair of words is then used to describe a horizontal sequence of 16 pixels.

Additionally, in order to shorten the above pointer to a width of eight visible pixels, it would be done by changing the third and fourth words of the SpriteImage Structure to look like this:

```
0000000000000000,0000000011111111
```

The leftmost eight bits of the word on the left, matched with the leftmost eight bits of the word on the right, all produce a "00" combination. The pixels corresponding to these bits will be rendered in color 16 which, according to the table above, is the transparent register. The remaining 8 pixels have a "10" binary combination, so register 18 will be used for these on the screen. This will produce a 16-pixel horizontal line in which only the eight pixels on the right will be visible.

Using binary numbers like this can get rather tedious, so lets switch to a shorter notation: Hexadecimal. In hex, each group of four digits is represented with a single hex digit, so 16 binary digits can be shortened to a four-digit hex number. In the hexadecimal system the numbers zero through nine are the same as for the decimal system. However, the numbers ten through fifteen are represented by the letters A through F. So twelve, in hex, is simply "C" or &HC in the AmigaBASIC notation. When converting binary numbers to hex, take each 4-bit group and substitute the hexadecimal equivalent. Translating the first example SpriteImage Structure above to hex, then, would look like this:

```
0,0
FFFF,FFFF
0,0
```

(Note: The AmigaBASIC prefix "&H"—which would be required for the interpreter to read them as hex numbers—is being omitted from these examples.)

The hex number "F" is fifteen or "1111" in binary, so 16 ones would be represented as "FFFF" in hex, and there's no need to write a hex "0000" to represent 16 binary zeros because zero is zero any way you look at it.

You may want a pointer that's a little fancier than a single horizontal line. To do this, just add more pairs of image data words. For example, a solid rectangle would look like this:

```
0,0
FFFF,FFFF
FFFF,FFFF
FFFF,FFFF
FFFF,FFFF
0,0
```

which represents a rectangle 16 pixels wide by four pixels high, rendered in the color of the 19th register.

Rendering more complex objects with this method takes a little practice. In the beginning it might be easier to draw the image first on a piece of graph paper within a width of 16 graph squares. Then take one horizontal line at a time and determine what color each pixel on that line should be. From this the two 16-bit binary numbers representing this line of pixels can be worked out to produce this scheme of pixels. Each binary number can then be converted to hex or, if preferred, decimal. Once this has been worked out for a few lines, the process will go much faster.

After the pointer's appearance has been determined, and the image has been translated into the appropriate numbers, the actual SpriteImage Structure in RAM must now be created. Then the SetPointer routine and its address can be provided. Placing this structure in RAM requires two steps.

First a suitable area in RAM must be found in which to place the structure. This is done with a routine in the exec library called AllocMem. Its syntax is:

```
addr&=AllocMem&(ByteSize,opt&)
```

Where addr& is a long integer variable of personal creation, ByteSize is the size of the desired memory block in bytes, and opt& is a value which defines certain options which will be discussed in a moment. When this routine is properly set up it will allocate the memory needed, and it will assign the address of this memory block to the addr& variable. Since this routine returns a value (the address of the allocated memory), it requires a DECLARE FUNCTION statement before using it. It looks like this:

```
DECLARE FUNCTION AllocMem&() LIBRARY
```

The value returned by AllocMem is a long integer, hence the "&" following the routine name.

The opt& variable tells AllocMem which of several options are desired for this memory allocation. In this case the value used is 65539 (2^0 + 2^1 + 2^16) which tells AllocMem to find us a block of stable memory within the area of RAM known as CHIP RAM, and clear all the bytes in this allocated area to zero. The SpriteImage Structure must be in CHIP RAM, because the structure contains image data. CHIP RAM is the lower 512K of RAM on original Amigas or the lower 1 meg of RAM on machines with the new Super Agnus chip, and it must be used for storing image data because it is the only part of RAM which the graphics chips can access. There are other possible options for use with the AllocMem function in different situations and a more complete description of these can be found in the "Amiga ROM Kernel Reference Manual: Exec" from Addison-Wesley.

As mentioned above, the ByteSize variable is the size of the desired memory block measured in bytes. The AllocMem routine always works with multiples of eight bytes, so it will round this figure up to the nearest multiple upon execution. After the required memory block is allocated, we actually create the SpriteImage Structure in RAM with POKEW statements. In doing this, the address returned by AllocMem is used as a point of reference. For example, if a pointer is going to be rendered as a single horizontal line (see first example), this is how the SpriteImage is created:

```
opt&=2^0+2^1+2^16
Pointer&=AllocMem&(12,opt&)
POKEW addr&+4,&HFFFF
POKEW addr&+6,&HFFFF
```

Here AllocMem is set up to allocate 12 bytes of memory, because the structure contains six words which is equal to 12 bytes. AllocMem will actually give us 16 bytes instead of 12, because 16 is the nearest multiple of eight bytes, but only the first 12 bytes are going to be used for the structure. None of the zero values have to be filled, because the AllocMem option was used, which already cleared the entire allocated memory block to zeros. Now simply use POKEW to insert both the image data words (&HFFFF), beginning with "addr&+4," because this data begins four bytes from the starting address of the structure. The above code completes construction of the SpriteImage Structure, now the Pointer& variable can be used as a parameter in the SetPointer routine.

One important point to keep in mind when using the AllocMem routine is that the memory it allocates remains allocated until it is specifically freed up or a reboot occurs. If the program does not free up this memory before it ends, the structure will remain in RAM after it ends, and waste system resources. The freeing-up of allocated memory is done with another exec routine called FreeMem. It's syntax is:

```
FreeMem& addr&,ByteSize
```

where addr& is the address of the memory block to be freed up, and ByteSize is its length in bytes. The EndIt routine in the accompanying listing shows an example of its use. One word of caution as regards to FreeMem: never attempt to use it to free up memory which was not first allocated by AllocMem or another memory allocation routine, as this will cause a system crash. Also, be sure that there are no typos in the addr& variable or other conditions that would cause the variable to default to a zero value, as this is also Guru-bait. If mysterious crashes occur while using this routine, check the code carefully for these two likely causes.

Let's review a few points here before going on. The maximum width of the pointer is 16 pixels. Each horizontal line of its image is represented by two words in the SpriteImage Structure. The bit combinations of each of these pairs of words determines the colors

of the pixels in each horizontal line. If a pointer is constructed that is, say, ten pixels high, the structure required to hold its complete definition will be ten words plus ten words plus four words, for a total of 24 words or 48 bytes. Remember, each line of the image requires a pair of words (four bytes) and must include the two zero words at the beginning and the two zero words at the end. The formula, then, for the length of the SpriteImage Structure in bytes is: (height*4)+8. This could add up to quite a few POKEW statements for any pointer more than a few pixels high. So, it usually saves some typing to set up pixel data as DATA statements and use a loop to READ and POKEW them into the structure. This is how it's done in the accompanying listing.

Do not use an array to hold the SpriteImage Structure and pass the array's address on to the SetPointer routine, because it will not work reliably. This was the first method I used when familiarizing myself with custom pointers, and the only result achieved was a completely invisible pointer! What's more, attempts to use an array for this purpose created the oddest glitch with the AmigaBASIC editor — the "ghost writer" bug.

After typing in code which used an array to hold the SpriteImage Structure and then running the program, I looked again at the original listing again and found that the editor had changed the source code! This happened time after time with several different code variations. The editor just didn't seem to like that array being there. Wherever it appeared in the listing, the editor removed it and replaced it with a continuous string of non-alphanumeric ASCII characters stretching out four-and-one-half full screen widths to the right! Moving the cursor to the beginning of this string, pressing the delete key once and then moving the cursor to another line caused the string to vanish. An attempt was made to run the program with the ASCII characters still there, and the program would just balk, producing a "Subscript out of range" error message. (This bug is still a mystery, but If anyone out there can enlighten me, please write.)

Getting back to the SetPointer routine, the last two parameters need some explanation. The XOffset and YOffset variables describe where the pointer's "hot spot" is wanted in relation to the pointer as a whole. This hot spot covers an area equal to the size of a single pixel and represents the screen location of the pointer as viewed by the system. So, the pointer's shape is really just a pretty package wrapped around the hot spot.

If both the X and YOffset values are set to zero, the hot spot will be default to the upper left corner of the pointer. Negative XOffset and YOffset values will move it to the right and down respectively. Positive values will do the opposite. The best place to put the hot spot depends on the shape of the custom pointer. With an arrow shape the tip of the arrow head is an obvious choice.

However, an arrow that points to the upper right instead of to the left would require the hot spot to be in the upper right corner of the pointer. In this case an XOffset of something like -15 would be used, while the YOffset would remain at zero.

If this business of negative x values moving the hot spot to the right and negative y values moving it downward seems backwards; that's because it is backwards. The hot spot itself doesn't really move, it's the pointer shape which moves in relation to it. A negative XOffset value moves the pointer to the left which shifts the hot spot rightward in relation to it. Likewise, a positive YOffset value will shift the pointer downward which gives the hot spot a higher relative position. Just remember that the offset values shift the body of the pointer in relation to the hot spot, and the negative/positive aspect will be easier to visualize.

That's really all there is to creating a custom pointer. First create the SpriteImage Structure and then call SetPointer, plugging in the appropriate parameters. The new pointer will appear immediately upon execution of SetPointer.

Now, if a custom pointer is created, does that mean the little default arrow is lost forever? Fear not. Intuition likes that little arrow far too much to let go of it that easily. It can quickly be retrieved with a simple call to the intuition library routine ClearPointer. Its syntax is:

ClearPointer W&

where W& is again a pointer to a Window Structure which can be obtained via the WINDOW(7) function. Its a good idea to always call this routine before exiting the program. If it is not used, the custom pointer will probably hang around after the program ends, until the user clicks the mouse on a window. The system will then reset the default pointer on its own.

The reason the default pointer is not lost is that an Intuition pointer is always linked to a specific window. Creating a custom pointer for one window does not affect the pointers of other windows. For example, the program opens three windows and window number three is currently active. A custom pointer is then created as outlined above, and it appears on the screen. If window number one or two then becomes the current output window, the pointer will revert to its default shape until window number three becomes active again, at which time the custom pointer will reappear.

This little wrinkle is easy to solve. Immediately after opening each window make a call to SetPointer, using the WINDOW(7) function as the first parameter. This function will always return the address of the current window, so the pointer will be attached to it when SetPointer is called. Then the code, for setting up a screen with three windows where all use the custom pointer, would look something like this:

```
SCREEN 1, ...etc
WINDOW 1, ...etc
SetPointer WINDOW(7),...etc
WINDOW 2, ...etc
SetPointer WINDOW(7),...etc
WINDOW 3, ...etc
SetPointer WINDOW(7),...etc
```

This ties the custom pointer to each of the three windows. No matter how the current output window is changed, either through source code or user clicks. The new Intuition pointer is maintained.

Can each window have a different custom pointer of its own? Yes—just make a separate SpriteImage Structure for each and include its address in separate SetPointer calls. Actually, as many pointers as desired can be created, or change at any time, with this routine.

The AmigaBASIC listing accompanying this article shows a simple demonstration of creating custom pointers. The first mouse click causes the default pointer to disappear, the second brings on a custom pointer and the third restores the default arrow.

The disappearing act is done by first defining a "dummy" SpriteImage Structure which contains only zeros and passing its address to the SetPointer routine. This creates an invisible pointer. It's actually still there, and it will function...but try to find it!

This invisible pointer trick actually has some utility as not all programs require a pointer. When not needed, a pointer can be a bit of a nuisance, so it's nice to be able to get rid of it when it serves no purpose and would only get in the way.

The Intuition pointer is an important part of the Amiga user interface, and it is also a tool often overlooked by programmers. A pointer which changes its shape to fit the current program option or mode is not only clever, but can be a useful reminder to the user as well. This would be true of a paint program in which the pointer assumes the shape of the currently-selected drawing tool, for example. A text message, written vertically, can also be communicated via the pointer using the appropriate image data. Animated pointers are also possible through the use of a series of SetPointer calls matched with SpriteImage Structures which define a sequence of image "frames" in page-flip fashion.

These are just a few, off-the-cuff ideas for applications of customized pointers. After a little experimentation, inventive readers will dream up many others.

Listing

```
'* * * * * * * * * * * * * * * * *
'*                               *
'*                               *
'* CUSTOM INTUITION POINTER DEMO *
'*                               *
'*     AmigaBASIC source code    *
'*               by              *
'*         Robert D'asto         *
'*                               *
'* * * * * * * * * * * * * * * * *

SCREEN 1,640,200,2,2
WINDOW 2,"Pointer Demo",,31,1

DECLARE FUNCTION AllocMem&() LIBRARY
LIBRARY "exec.library"
LIBRARY "intuition.library"

CLS:PRINT
PRINT "Click mouse to make pointer disappear."
WaitClick

NoPointer

CLS:PRINT
PRINT "Click mouse again to create custom pointer."
WaitClick

CustomPointer

CLS:PRINT
PRINT "Click mouse again to restore default pointer"
PRINT "and end program."
WaitClick

ClearPointer WINDOW(7)

EndIt:
    '* free up the memory allocated earlier
    '* for the SpriteImage structures
  FreeMem p&,76
  FreeMem np&,8
  LIBRARY CLOSE
  SCREEN CLOSE 1
  END

SUB NoPointer STATIC
    '* creates SpriteImage Struct and
    '* pointer with no visible image data
  SHARED np&
  opt&=2^0+2^1+2^16
  np&=AllocMem&(8,opt&)
  SetPointer WINDOW(7),np&,1,1,0,0
END SUB


SUB CustomPointer STATIC
    '* declare a SpriteImage address
    '* variable as SHARED so we can
    '* free up the memory later in
    '* EndIt routine
  SHARED p&

    '* allocate memory for the structure
  opt&=2^0+2^1+2^16
  p&=AllocMem&(76,opt&)

    '* fill in the image data  beginning
    '* with an offset of 4 bytes from the
    '* beginning of the structure
  FOR x%=4 TO 66 STEP 2
    READ d$
    pixeldata%=VAL("&H"+d$)
    POKEW p& + x%, pixeldata%
  NEXT

  DATA 1FF8,1FF8,1FF8,1FF8,0,1FF8
  DATA FFFF,FFFF,3FFC,0,7BBE,440
  DATA 7FFE,0,FFFF,0,FFFF,0
  DATA 6EEE,100,77DE,0,383C,0
  DATA 1FF8,0,7E0,0,FF0,E70
  DATA 3E7C,3FFC

    '* give all the parameters to the
    '* SetPointer routine
  SetPointer& WINDOW(7),p&, 16, 16, 0, 0
END SUB

SUB WaitClick STATIC
  WHILE MOUSE (0)<>0
  WEND
  WHILE MOUSE (0)=0
  WEND
END SUB
```

•AC•

BATMAN: The Movie

"BATMAN: THE MOVIE" IS A NEW action adventure game from Data East. In it you assume the title role of Batman. Since the game is based on the most recent Batman movie, this game doesn't involve Robin, the Boy Wonder. The game sticks to the plot of the movie rather faithfully, highlighting the best action sequences for use in the game.

There are a total of five scenarios through which you must guide The Dark Knight. The first game sequence recreates the shootout in the Axis Chemical plant, in which Batman shoots Jack Napier. This unfortunate event causes Jack to fall in a vat of toxic chemicals, thus creating the Joker. Your only weapons in this sequence are the Batarang and the Batrope. You throw the Batarang at the villains throughout the chemical plant to eliminate them, while they fire at you with their grenades and pistols. As your health declines, your face at the bottom of the screen slowly turns to that of the Joker. Besides dodging bullets, you must also avoid the spray of toxic chemicals throughout the plant. The Batrope is used to swing Batman from level to level, in his search for Jack Napier. You have a total of 8 minutes to accomplish this mission.

At the end of the first level Jack is transformed into the Joker, who in the second level chases you through the streets of Gotham City. Your vehicle is the infamous Batmobile, equipped with a computer which shows you the correct route to the Batcave (See picture). The computer displays an arrow at the top of the screen to show you at which streets you must turn. You only get three chances to turn, after which you will run into a police road block if you are unsuccessful. Since the Batmobile moves at such great velocities, you must engage the use of a Batrope to grab hold of the light poles at the intersections in order to make a turn. Hitting any obstacles along your 100 block path

Batmobile enroute to the Batcave.

THE JETSONS

ALL RIGHT, SO MAYBE I'M A KID AT heart, but I enjoy watching cartoons. Of course, not just any cartoon, usually the old classics - Johnny Quest, Bugs Bunny, and THE JETSONS! It was thus with much glee that I brought home "George Jetson and the Legend of Robotopia" Could they actually put the look and feel of a cartoon into a computer game? I was about to find out!

Greetings from the planet Robotopia.

In "The Jetsons", you become George Jetson, head of the Jetson household. Along with this honor, you get to perform the usual fatherly duties such as hunting for your car keys after your teenage daughter has borrowed the car; attending school science fairs; and going to work. Mr. Spacely is still bossing George around. Only this time George may deserve it - he's two hours late for work, and Spacely is ready to fire him. The only thing that George can do to make up for his tardiness is to accept a special assignment. It seems that Spacely has invested millions of dollars in a resort on what was the beautiful planet of Robotopia. To finish the project, he needs more funds from a group of large investors who are headed to Robotopia to check the resort out. The only hitch is that the robotic residents of the planet seem to be at war, and thus the planet is a disaster. Spacely wants you to find out what is going on and correct it before the investors arrive. If George is successful, he will not only keep his job, but get a hefty bonus as well.

The game is provided on two disks, along with a cartoon manual which provides the history of what happened the day before the game starts. This manual is also the source for words you must look up during the game as a form of copy protection. Luckily you only have to do this once. The manual also explains to the player how the game interface works.

The game interface is extremely well done - especially for someone like myself who usually doesn't like adventure games. It is completely mouse driven - which also means you don't have to "guess what word the programmer wants you to use to get by this puzzle". The upper left portion of the screen is George's view of the area he is presently in. Below this is a text window, in which the description of your surroundings appear, as well as your conversations with other characters. You speak with other characters by selecting a response in the window immediately to the right of the text window. It's your job to choose the correct response. If you try to think like George Jetson, you'll do all right.

The interface works well, and is very easy to use. The View Screen is actually animated for some portions, which really adds to the game. The program also uses digitized sounds from the show, which also adds to the ambience. There are many characters with whom you can interact, and the plot of the game is interesting yet easy to follow.

There is only one pull down menu, which allows you to restart a game, save or restore a game, or quit a game in progress.

(continued top of page 57)

*(**Batman**, continued)*

will drain your life, and waste time. You only have 5 minutes to reach the Batcave to begin work on deciphering the Smilex mystery.

It seems the Joker is poisoning people all over Gotham City with a chemical called Smilex. He puts the ingredients in 3 separate cosmetic agents, which when combined form the deadly Smilex poison. Using the Batcomputer, Batman must find out which products contain the poison. The Batcomputer has narrowed the choices down to eight chemicals. Using the Batcursor, you combine three of these at a time to see if any contain the Smilex. The computer responds by telling you how many of the three items selected contain Smilex. This is very similar to the game of Mastermind, but with two caveats: you only have 6 "tries", and 30 seconds in which to solve the puzzle.

Batman has succeeded in outsmarting the Joker at every turn so far, but now the Joker plans to destroy the population of Gotham City by spraying them with poisonous gas hidden in parade balloons at the Gotham City Carnival. You as Batman rush to the Batplane, and fly down the streets of Gotham snapping the tether lines leading to the balloons, allowing them to float to safety. Time is of the essence, as you have calculated that you must release 100 balloons within 5 minutes in order to thwart the Joker's plan.

The last mission is the most difficult. The Joker has taken refuge in the Gotham City Cathedral, awaiting the arrival of his helicopter to flee to safety. Batman has only 12 minutes to find and destroy the Joker, once and for all. This scenario is very similar to the Axis Chemical plant, in that all you have to work with is your Batarang and Batrope. There are more goons in this level, equipped with automatic weapons and more grenades. You must also watch out for floors which collapse beneath your feet, which may lead to your ultimate demise (and thus the demise of Gotham City).

The playing screen is divided into two main portions, the action screen (at the top) and the statistic screen (at the bottom). The statistic screen shows your current and high score, your health (the fading picture of Batman), the time remaining for the level you're on, and how many lives you have left. You are granted 3 lives for the entire game, but can earn an extra life for every 100,000 points scored. The game is provided on one disk (copy protected) with a well done, 6 page instruction manual.

BATMAN features well done sounds and graphics. You can see as well as hear the Batmobile burn rubber, and watch the Batplane as it is slowly consumed by flames. My game had an inexplicable yellow line which appeared at the upper left corner of the screen and seemed to move around, but it never bothered me during game play. Overall, I was quite impressed with the smooth scrolling screens and excellent sound effects.

The programmers wisely included a pause key (F1), as well as key (F2) to turn on and off the soundtrack. For some reason, you aren't allowed to pause the game during the 30 second Batcave sequence. Unfortunately, there is no way to save a game in progress, nor are the high scores saved to disk. Another feature I would have liked to see is the ability to skip a level if you have already completed it. As it stands now, if you lose on Level 5, you must replay all 4 of the preceding levels.

"BATMAN" is a fun and exciting rendition of the movie of the same name. Action is present in quantity, with an exciting musical score. It's one of those games that keeps you coming back for more. If you've ever dreamt of becoming the Caped Crusader, try "BATMAN". You WON'T be disappointed. —*Miguel Mulet*

ADVENTURES THROUGH TIME
Vol. 1: The Scavenger Hunt

A TIME MACHINE! I DON'T KNOW anybody who wouldn't want access to one for just a little while. Just think - you could go back and correct past mistakes, or maybe venture into the future and bring back information that would help you in the present. Alas, mucking with time could be dangerous, and it's doubtful that you or I would ever get to see the inside of a time machine. Young Buck Walker, however, is luckier than the both of us. It seems that in the 21st century, time travel is possible - and it is a closely guarded secret. Only Historians of the United Earth - an organization which in 2059 AD oversees all the nations of Earth - have access to a time machine. Buck's father is one such Historian, who happens to be away attending a national convention of Historians. That leaves Buck and his rebellious friends unsupervised, and when the cat's away, the mice will play!

In "Adventures Through Time: The Scavenger Hunt", an adventure game from Aurum Software, you become Buck Walker, the rebellious young son of a time traveling Historian. Time travel, until this time, had been used to record history accurately by actually observing past events. It seems Buck and his friends (whose parents are all Historians) have decided to go on a scavenger hunt through time - collecting relics from the past. These relics include a lyre, a tomahawk, a shield, a dinosaur egg, and a papyrus scroll. The kids don't mean any harm, it's just that as children of Historians they aren't allowed to play with other children or each other (the government feels this would compromise the security surrounding time travel). Thus, Ace, Daisy, Bookworm, and Buck resort to this "harmless" jaunt through time in order to amuse themselves.

"Adventures Through Time" is provided on a single non-copy protected disk, and includes a well written 22 page manual, an "Historian's Handbook" to time travel, a United Earth decal, and a "Time Machine Operators License". To start the game, you first must make a backup copy of the game. Following this, just boot up the machine with the backup at the Workbench prompt. Have your Workbench disk (1.2 or later) handy, as the first time you load the game certain files are copied from the Workbench disk to the game disk. The game will run in 512K. While the disk is not copy protected, you will need the Historian's Handbook (printed on that difficult to read maroon paper with black ink) in order to type in the appropriate word to access the time machine.

After the game has loaded, you are given a brief introduction and then you're ready to play. The game is in real time, meaning that for each minute of real time which passes, one minute passes in the game. (A pause feature is provided). You start in Buck's room, which is illustrated in the top half of the screen. In the middle of the screen is a status line which shows your condition, your score, the time of day, and the year (a useful feature when you're traveling through time). When the right mouse button is depressed, the status line changes to a menu bar with three headings: GAME, PREFS, and HELP. The game menu allows you to load and save up to seven different games, as well as pause, quit, or restart the game. The PREFS menu gives the ability to change the text size, skill level, or drop into the Workbench. The HELP menu offers help in playing the game.

Below the status line is the narrative window and a control panel. Commands are typed in the narrative window, which also serves to display descriptions of items, places, and conversations. At the lower right of the screen is the control panel which helps decrease, but not eliminate, typing. To move the character left or right, you hit the left or right gadgets. To open an item, you press the OPEN gadget and then point to the item on the screen you wish to open. This only works if the item is shown on the screen. If the item is not on the screen, you can click on the word in the narration window if it is present there. I found that most of the time it was easier just to type in the commands rather than use the control panel. Typing is made a

*(**The Jetsons**, continued from page 55)*

Unfortunately, I did have a couple of major problems with the game which almost prevented this review. The first problem was that the game refused to boot at the Workbench prompt. It took me a while to locate the problem - which happened to be my Amiga 1300 Genlock. A quick call to MicroIllusions confirmed that for unknown reasons, "The Jetsons" just doesn't like genlocks (of any sort). They said that the only workaround was to completely disconnect the genlock from the machine. As much as I hate to connect and disconnect equipment just to run a game, I still felt that it was worth the hassle.

The next problem was a little more severe. After playing the game for 2 hours, I got a visit from the guru. The error given was a Read/Write Error on Disk #2. "No problem," I thought. I'll just restart it from where I last saved the game. Unfortunately, I got the same error on trying to restore the game. I started the game completely over (ie, a cold reboot), but the game again crashed at exactly the same place. At this point, I went back to the store and got a second copy. I got the same error in the same place with the second copy too! So, back to the store for copy number 3, but it was to no avail. After going through three copies, I felt that maybe my machine was at fault (despite the fact that no other program I owned was crashing). Thus the Amiga was put through an intensive set of system and disk diagnostics which detected no problems. As a last ditch attempt (I really did like playing the game when it worked), I sent the disks back to MicroIllusions, who returned them two weeks later with the game copied onto them. Finally, I had a working copy of the game. MicroIllusions stated that they had never had this problem before. I'm not sure if my area was just unlucky and got a bad batch of disks, but the problem was most infuriating.

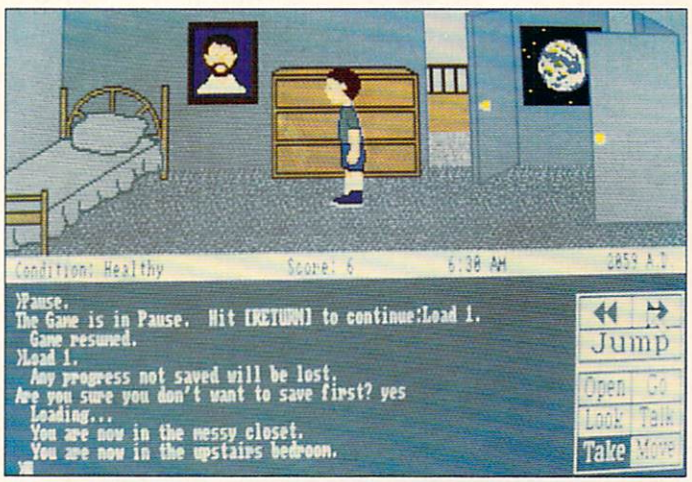
Overall, I really enjoyed playing "The Jetsons". The interface is well done, as are the sound effects and graphics. The plot is well thought out, and keeps you interested in playing the game (and completing the adventure). As long as you don't have the read/write errors I encountered (and don't mind disconnecting your genlock), I can recommend this game. I would definitely try the game at your local computer store before you take it home.
—Miguel Mulet

BATMAN: THE MOVIE
Game Design by Ocean Software
Distributed by Data East USA, Inc.
1850 Little Orchard Street
San Jose, CA 95125
List Price: $39.95
Inquiry #209

The Jetsons
MicroIllusions
17408 Chatsworth St.
Granada Hills, CA 91344
1-800-522-2041
Price: $44.95
Inquiry #208

Adventures Through Time
Vol. I The Scavenger Hunt
Aurum Software
P.O. Box 5392
Ventura, CA 93003
(805) 659-3570
Price $49.95
Inquiry # 210

Young Buck in his room.



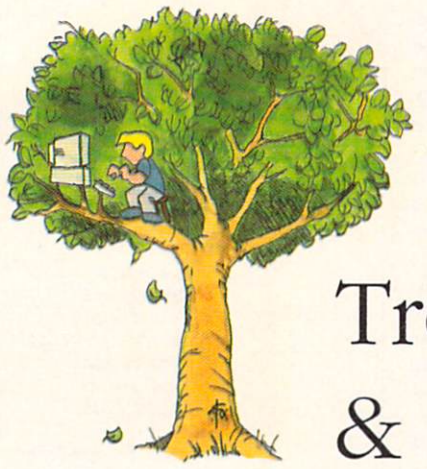
little easier because the programmers use the function keys as substitutes for commonly used commands. (Thus the F1 key automatically types LOOK for you). A template is provided to put on top of the function keys so you don't have to memorize the commands.

Buck has several obstacles to overcome in order to win the scavenger hunt. First of all, he must find where the time machine is hidden. Provided he does this, he must then discover how to operate the time machine. Lastly, he must travel to the specific eras in which the items can be found, and convince their original owners to part with the possessions. These tasks are fairly straight forward, once the right sequence of events are followed. The fun of the game is in discovering how each of these tasks can be accomplished.

Each of the five eras are fairly easy to get around in. Most of them are linear, ie, there is only one main path with a few rooms to explore off to the side. In the prehistoric and western worlds, you're required to jump over a few obstacles which gets tedious after a short while. Ancient Greece is perhaps the smallest world, but one of the harder eras in which Buck must accomplish his task. The pyramid is the hardest level to get around in, as it is a veritable maze of different passages. You'll need to dig out the pencil and paper to map that world out.

As far as adventure games go, "Adventures Through Time" lacks many of the niceties that current adventure games offer. For example, many games completely eliminate typing—allowing the player to concentrate on the game rather than the phrasing of a certain command. The parser used in this game does not help much either - its vocabulary is extremely limited, meaning the player may have to try a lot of different wordings before the computer will understand what to do. Also, in most graphics adventures the objects that can be found in a room are displayed on the screen. In "Adventures", the player must rely exclusively on the narrative to see what items have been discovered. Since the objects are not displayed on the screen, the player cannot just pick up the item by moving over the item and clicking. Lastly, the game features limited graphics and NO sound (not a peep!). There are a few animations during the game The graphics are very simple, and screen scrolling gets choppy in the few scenes where graphics are a little more complex.

Since "Adventures Through Time" doesn't make good use of Amiga sound or graphics capabilities, it relies heavily on the plot and game play. Unfortunately, these are only fair as far as adventure games go. Maybe I'm spoiled, but I like the "no typing" and "heavily graphics oriented" features that are offered by other adventure games that are currently available. Most other games also make liberal use of sound effects and soundtracks. "Adventure Through Time" is "cute" as is, but is more likely to appeal to a younger crowd. *—Miguel Mulet*



Tree Traversal & Tree Search

by Forest W. Arnold

IN MY LAST ARTICLE, "TREES AND RECURSION" (AC V4.12), I WROTE about natural binary trees and showed how to build the trees as a list of lists. Natural binary trees are very useful data structures, but you need to know how to move around in the trees and to find whatever data is stored in the nodes in order to put them to work.

I'll discuss two common methods for traversing trees. Then I will explain how the two traversal methods can be used to search for a node in a tree. Before diving right into a discussion of tree traversal and search techniques, I'll briefly review what natural binary trees are.

NATURAL BINARY TREES REVISITED

Figure 1 shows the tree which was discussed in my last article, and figure 2 shows how the tree is implemented as a list of lists using only two pointers in each node. One pointer is directed at a node's sibling node. This is the standard 'next' link pointer in a linked list. The other pointer points to a linked list containing that node's child's node. The arrows in figure 2 which point to the right represent the sibling link pointers, and the arrows which point downward represent the child link pointers. The top node, numbered 0.0, is called the root node of the tree, and the nodes numbered 3.0 through 3.7 are called leaf nodes. The tree has four levels, numbered 0 through 3. The root node is on level 0, and the leaf nodes are on level 3. The C structure defining a node in the tree is:

```
typedef struct cleverNode

{

   struct cleverNode  *next;  /*  sibling link
pointer */

   struct cleverNode  *child;       /*  child
link pointer   */

   unsigned char      *data;        /*   this
node's data     */

} CLEVER_NODE_T;
```

Trees constructed as lists of lists using this type of node definition are called natural binary trees. With this single clever node definition, just about any type of hierarchical structure can be represented in a C program. By storing function pointers in the tree nodes, even C programs can be constructed as natural binary trees! In my last article, I showed you how to build the tree. But to use the tree once you've succeeded in building it, you need to know how to move around from node to node. Here's how it's done...

TREE TRAVERSAL

To find a node in a tree, you not only need some method for moving from one node to another, but you also need a method which will allow all of the nodes to be visited, while not

allowing the same node to be visited more than once. All of the nodes in a list of nodes connected by 'next' pointers can be visited by starting at the first node in the list and following the 'next' pointers to the last node. This is the standard way linked lists are traversed. It is very easy to visit all of the nodes on a level which are connected to each other. Recursion can be used to travel between levels of the tree. Recursive procedure calls are used to move down the tree, and procedure returns are used to move back up to the previous level. By combining recursion and iteration using 'next' pointers, you can get to all levels of the tree and to all the nodes in a list. But how do you know for sure that all nodes will be visited (once)?

A path in a tree consists of a sequence of linked nodes. In figure 2, one path consists of the connected nodes 0.0, 1.0, 2.0, 3.0. All of the links in a tree are 'directed', which means they only go in one direction. This means that any path from a node to any other node will also be directed. Since all of the nodes in a tree are connected to each other through some path and all paths go from any node to a leaf node, any path in a finite tree is guaranteed to end and guaranteed not to double back on itself. And since all of the nodes are connected through a path, every node in the tree can be visited by starting a path at the root node and visiting all the children of the root, all of the children's children, etc. What all of this means is that with the right method, you can be assured that all nodes in a tree can be visited, and that each one will only be visited once. Visiting the nodes in a tree is called tree traversal.

Two common traversal techniques for trees are depth-first traversal and breadth-first traversal. Many algorithms manipulate trees by using one or both of these methods. Almost every procedure in the demonstration program uses depth-first traversal. The ideas behind each of these two traversal techniques are simple.

In depth-first traversal, nodes are visited by following the child links first, moving through the tree from top to bottom. When a leaf node is found, sibling links are followed from left to right until the end of the sibling list is found. The traversal then returns to the parent node of the leaf, and the parent's sibling links are followed from left to right. If any node in a sibling list has a child node, the child link is followed. This process continues until the traversal recursively returns to the root node and ends. The traversal rules are:

Rule 1: If the current node has a child link, follow it.

Rule 2: If the current node does not have a child link or the child was already visited, follow the sibling link, if there is one.

Rule 3: If the node does not have a sibling link or all sibling nodes have been visited, move back up the tree to the parent node.

Figure 3 shows the order in which nodes are visited during a depth-first traversal. Given the structure of a natural binary tree, depth-first traversal is straight-forward: if a node has a child node, visit the child; otherwise, visit the sibling. Child nodes are visited by recursion and sibling nodes are visited by following 'next' pointers. A recursive C procedure to perform depth-first traversal is:

```
void dfTraverse( topNode )
CLEVER_NODE_T       *topNode;
{
   CLEVER_NODE_T *node;

   node = topNode;
   while( node )
   {
      if ( node->child )
         dfTraverse( node->child );   /*  recurse       */
      node = node->next;              /*  iterate       */
   }
}
```

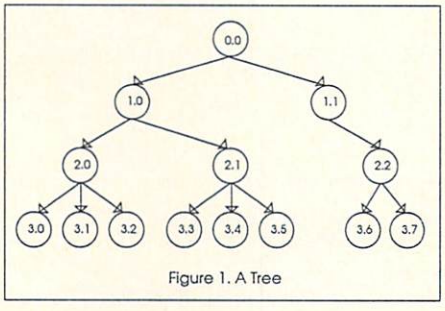


Figure 1. A Tree

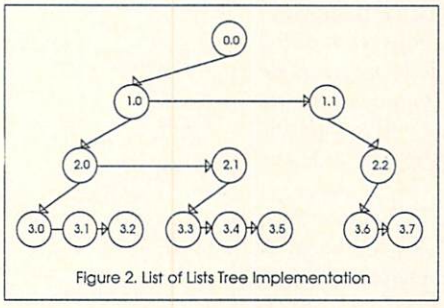


Figure 2. List of Lists Tree Implementation

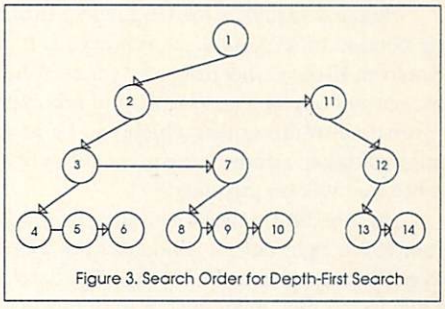


Figure 3. Search Order for Depth-First Search

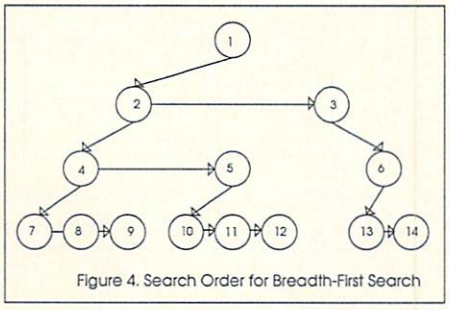


Figure 4. Search Order for Breadth-First Search

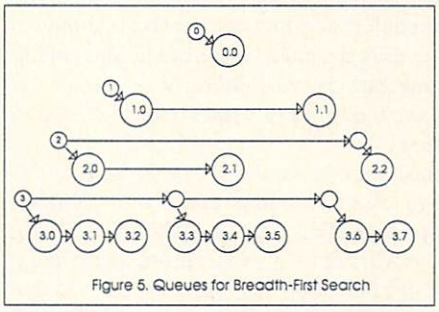


Figure 5. Queues for Breadth-First Search

The 'while' loop implements the iteration on the sibling list by following 'next' pointers until a node which has no 'next' link is reached. The loop then falls through and the procedure returns, traversing back up the tree. Within the loop, child links are traversed by recursion before the 'next' sibling node is visited. When a node without a child is reached, its sibling is visited, and the process continues until all reachable nodes have been visited.

Recursion makes the language perform all of the bookkeeping needed to remember parent nodes. It also keeps track of the data from all the earlier traversed states. When called with any node in a natural binary tree as the input argument, this procedure will traverse all of the nodes which can be reached by following the links leaving the node. If the initial input node is the root node, the entire tree will be traversed.

In breadth-first traversal, all of the nodes on a level are visited from left to right before child links are followed down to the next level. The traversal order is left to right across an entire level, then down to the next level. Just as in depth-first traversal, recursion is used to move between levels, and iteration is used to visit all of the nodes on a level. The main difference between the two techniques is that depth-first traversal visits the child nodes the before sibling nodes, and breadth-first traversal visits the sibling nodes before the child nodes. Figure 4 shows the order in which nodes are visited during breadth-first traversal.

Breadth-first traversal sounds as simple as depth-first traversal, but it is actually much more complicated to implement, because all of the nodes on a level may not be in the same sibling list. If this is the case, there will not be a 'next' pointer connecting the last node in one list to the first node in another list on the same level. The bottom two levels of the tree in figure 2 show this. It's real easy to visit node 2.0, then 2.1, but how can node 2.2 be reached from node 2.1? The same question applies to the bottom level. How can node 3.3 be visited after node 3.2 is visited, and how can node 3.6 be reached from node 3.5?

What is needed is some way to link all of the lists on the same level together, so that when the traversal reaches the end of one list, it can jump to the first node in the next list. This is just the type of problem which can be solved with queues (remember that a queue can be implemented in C as a list with new nodes added to the end of the list). As the nodes are visited from left to right on one level, the child lists on the next level are put in a queue, then the queue is used to traverse the nodes on the lower level. Figure 5 shows how the child lists are linked together in a queue. The small circles represent the queue nodes. The queue numbered 0 is the start queue. The queue for level 1 is built as level 0 is being traversed, the queue for level 2 is built as level 1 is being traversed, etc.

To start the traversal, a queue consisting of the top node is built, and traversal proceeds using the following rules:

Rule 1: If the node has a child list, put a pointer to the child list in a queue.

Rule 2: If the node has a sibling, visit the sibling.

Rule 3: If all nodes on a level have been visited, recurse using the queue to traverse the next level.

Building the queue is fairly easy. The trick is to build the queue at the same time the breadth-first traversal is happening. Naturally, I have an algorithm up my disk drive for doing just this trick, although it's neither pretty or efficient. The queue node structure is:

```
typedef struct queNode

{

   struct queNode *next;          /*  link to next node         */

   CLEVER_NODE_T  *nodeList;  /*  list of sibling nodes */

} QUENODE_T;
```

The C procedure for performing breadth-first traversal of a natural binary tree is:

```
void bfTraverse( queue )
QUENODE_T *queue;
{
   QUENODE_T  *topQNode,*newQNode,*prevQNode,*nextQNode;
   CLEVER_NODE_T   *topNode;
   topQNode = prevQNode = NULL;  /* next level's queue     */
   while( queue )                /* iterate on queue       */
   {
      topNode = queue->nodelist; /* get sibling list       */
      while( topNode )           /* iterate on list nodes */
      {
         if ( topNode->child )   /* put childlist in the  */
          {                                /* queue of child lists  */
            newQNode= (QUENODE_T*)malloc(sizeof(QUENODE_T));
            newQNode->nodeList = topNode->child;
            newQNode->next = NULL;
            if ( prevQNode )
               prevQNode->next = newQNode;
            else
               topQNode = newQNode;
            prevQNode = newQNode;
         }
         topNode = topNode->next; /* get next sibling node*/
      }
      nextQNode = queue->next;    /* get the next queue    */
      free( queue );              /* free current node     */
      queue = nextQNode;
   }
   if ( topQNode )                /* If a queue was built,*/
      bfTraverse( topQNode );     /* recurse               */
}
```

(continued on page 79)

ROOMERS

by The Bandito

[The statements and projections presented in "Roomers" are rumors in the purest sense. The bits of information are gathered by a third party source from whispers inside the industry. At press time, they remain unconfirmed and are printed for entertainment value only. Accordingly, the staff and associates of* Amazing Computing™ *cannot be held responsible for the reports made in this column.]

COMMODORE HIRES ATARI'S FOUNDER

Yes, that's right, sports fans. Commodore International Ltd. announced that Nolan Bushnell (inventor of Atari and Chuck E. Cheese's Pizza Time Theater) has been hired as general manager of Commodore's Consumer Interactive Products.

So what is this division, you ask? Well, this is the part of the company that's developing (you guessed it) the Amiga CD-ROM player, though of course Commodore didn't officially say that. What they did say was interesting: They announced that Nolan is already at work with his team on a product they expect to demonstrate at the June Consumer Electronics Show in Chicago.

So it looks like Nolan, who is a savvy guy with plenty of other things to keep him busy (half a dozen companies that he either set up or is heavily involved with running), has been seduced by something so cool that it made him give up the life of the independent entrepeneur and go to work for a big company. It must be that Nolan really believes in the potential for the Amiga CD-ROM player to be a hot piece of consumer electronics. And this is a guy who's seen quite a few technological wonder toys in his time.

The other interesting "tidbyte" from this announcement is that Commodore may be showing the Amiga CD-ROM openly at CES, rather than behind closed doors. They might even have plans to ship it for the Christmas season, rather than waiting until summer of 1991. The Bandito has managed to cobble together a few more facts about this dream machine. It's a slick black device that looks like part of your stereo on the outside, but it's a true Amiga 500 on the inside. We're talking 1 megabyte of memory, the new chip set, Workbench 2.0, and a full complement of ports. The Bandito hears about slick black peripherals that use infrared connections instead of cables. Joysticks, mice, even a keyboard that can be used from that couch you keep in front of the entertainment center.

SOFTWARE TURNS SOFTER DEPT.

Mediagenic is in big trouble lately. Fiscal year 1990 results are in, and they look somewhat disappointing. What would you say if your company lost $19 million dollars on $65 million in sales? You might say that you have a problem. What's caused all this? A combination of factors. Mediagenic has always had high expenses, and their sales slowed down in several areas while their expenses didn't. Then in April, Mediagenic lost a patent infringement suit filed in 1986 by Magnavox; the court ruled that Mediagenic was distributing an Atari video game on which Magnavox had a patent.

Mediagenic made some decisions to discontinue several divisions (Infocom, their Triton mail order business, and their Apple Presentation Tools).

[Editor's note: Mediagenic has confirmed they worked out an agreement with Magnavox regarding the infringement suit.]

MORE COMMODORE NEWS

Here's another interesting tidbit that the Bandito picked up while haunting the electronic cocktail parties. Commodore officials are wooing Prodigy, the IBM/Sears telecommunications service, in an effort to get the service to appear on your favorite machine. Look for an announcement this summer if they can put together a deal. While Prodigy is slow, it is graphics-based and very showy. The Bandito suspects that the software might take advantage of the Amiga's sound capabilities as well as graphics, to provide a very interesting and entertaining service. Imagine advertising with theme music and sound effects!

The Bandito hears that Commodore is also interested in other telecommunications services to provide Amiga-specific support with intelligent, graphical interfaces that make telecommunicating easy. And why not? The Amiga should be able to do that better than any other computer.

Mass-market Amigas are coming soon to a major retailer near you. After many months of maneuvering and deciding, Commodore has finally determined to take the plunge. Look for announcements this summer of availability for the Christmas buying season. Commodore has closed deals with some major retailers already, the Bandito hears. The A500 will hit with a street price of considerably under $500, or so the rumors go. The prospect has software developers optimistic, especially if their software can be among the few titles carried by the mass market stores. Look for a heavy concentration on games, with maybe a paint program and a word processor thrown in for good measure. Oh, and some educational titles, too.

So what's left for the computer retail store? Plenty. The retailers can provide all those add-ons and options that the mass market won't deal with, like the A590. And there's a good price drop in store for the A2000 series once the A3000 hits the stores, which is even better news for the retailers. How does a list price of $1499 sound for an A2000? Of course, the street price will be even lower.

Will the A2000's get the spiffy new A3000 case? Maybe not, but there is a new version of a case for the A2000 series in the works. Look for it to appear around Christmas, if it can ever get approved.

SON OF WORKBENCH DEPT.

Workbench 2.0 appears to be generating even more excitement than the A3000, among those familiar with both. Very slick, according to those who have played with it. There's still a number of bugs to be squashed, but by the time it comes out the software should be very stable. Developers seem to think that revving their software will be fairly straightforward, even for those who break a lot of rules. So save some dollars, because you'll be upgrading a lot this fall.

Workbench 2.0 has a lot of very nice features. For instance, there are no more invisible files (a default icon is provided). You can create a virtual size screen up to the limits of chip RAM, and in the case of the A3000, that means a screen that could be thousands of pixels in both dimensions. But the Bandito thinks the much cleaner graphics and the easy way you can customize the fonts and graphics are what's most important. The visual look, especially on a 640 x 480 non-interlaced screen, is stunning. It's the equal of any other computer, and finally gets away from the rather clunky lo-res look of the original Workbench. Appearance counts for a lot in this business, and now the Amiga will look like the serious computer it really is.

Speaking of serious, the A3000 has a number of technical advances that aren't readily apparent, but will become more important over the next few years as a basis for future advances in the Amiga architecture. For instance, writing data to chip and fast RAM is more than twice as fast as the A2500/30 because of the 32 bit wide data path; this makes the video speed faster than the fastest new Macintosh, and four times as fast as standard VGA speed on an IBM. Many of the new chips are CMOS construction for high speed and low power consumption; eventually all of the custom chips will be CMOS, which will make for even greater speed (not to mention it makes a laptop Amiga easier to do).

The new architecture and Workbench 2.0 make it possible to create video cards with greater resolution than the standard Amiga resolutions. In fact, some boards are already in the works. The Bandito hears of a board from the U.K. (codenamed Hi-Tension) that puts out 1600 x 1200 pixels, with 256 colors out of 16 million. By putting a coprocessor on the add-in video board, given the speed of the new bus, it's possible to get the Amiga performance we're used to while still displaying thousands of colors at very high resolutions.

Commodore is also working in deep, dark secrecy on a new sound chip, or actually a combination of chips at this point. There's no plan yet on how it will be brought to market, but look for full 16-bit sound, digital signal processing capability, and the capacity for compact disc quality audio.

A3000 has better game compatibility than the A2500, according to those who have played with it. Unlike the A2500/30, though, you can't switch to a 68000 because there is no 68000 around. So if you're a game player, you may have to give up some of your old favorites if you buy an A3000.

Hey, Commodore, good job on the A3000 three page ad in the Wall Street Journal. It was very well executed, and made a good impression on those business types. Do that again, says the Bandito.

The A3000 is turning some heads at major software publishers. Hmm, since Commodore has worked out a deal with Novell for their network software on the Amiga, and Novell is merging with Lotus, do you think that...well, we'll see. Talks are continuing.

So why is Commodore's stock down at 7 with all this good news? The Bandito doesn't know. Maybe it's because the takeover fever has subsided. Or, that the hardware market overall is slow. But even the pending relaxation of COCOM restrictions, which means that Amigas will soon be sold in Eastern Europe has failed to boost the stock very much. Go figure.

THE NEXT AMIGA DEPT.

The engineers are hard at work on yet another Amiga, tentatively known as the A3500, that will feature a 68040 as the CPU and a 1.76-megabyte high density floppy that's backward compatible with the current floppies. The ability to read and write IBM and Mac disks will be easy to implement; in fact, this capability may go into Workbench 3.0, along with virtual memory and other goodies. Workbench 3.0 right now is just a list, but as the programmers finish up with bug-smashing for Workbench 2.0 they will be moving on to 3.0.

AN APPLE FOR THE AMIGA?

Stick with the Bandito on this item; there's an Amiga connection coming up. Here's how it goes: Just when you thought that the Apple II was deader than Rob Lowe's film career, it rises again. Why? Because of the stranglehold that Apple II software has on the very large education market. The schools don't have all that much money to spend, so they want to keep using the software they already have. But on the other hand, they do want to move up to new machines (though they are very sensitive to high prices). And the computer manufacturers realize that when Johnny uses a computer at school, Mommy and Daddy may buy him a similar one at home. All this is causing a huge battle for the education market, with Apple losing market share rapidly due to their overpriced, underpowered, nonpromoted Apple II line.

IBM has just announced some new computers aimed squarely at the school market. And to sweeten the deal for IBM, Big Blue has just bought a company that makes an Apple IIe clone on a card for IBMs; this card will sell for around $200 by IBM. So IBM is hoping that this will convince schools to go for IBMs with spiffy color graphics and an industry standard operating system, instead of Apples with no color (Macs) or ugly color (Apple IIs).

Meanwhile, Apple is trying hard to sell Macintoshes to schools. They just lowered the price to schools for low-end Macintoshes, providing a stunning 66% discount off the list price. So Mac Plus would sell for about $600, which is only a little more than it is actually worth. This is in anticipation of their low-cost Mac introduction in October; that machine will be priced even lower to schools. And Apple is talking about an Apple II emulator in software (a hardware Apple II emulator for more expensive Macs is also in the works). All this emulation is designed to attract the education market, which is hip-deep in old Apple II software. They don't want to spend money on both new hardware and new software, so Apple II emulation is important to them.

Enter into all this Commodore. They have by far the best computer for schools: low priced, great color graphics, animation, sound, an easy-to-use operating system, and some pretty cool software. But wait! No Apple II compatibility, so the school can't use all that Apple II software. Ah, but the answer is on its way: an Amiga developer already known for emulation is said to be coming out with an Apple IIe emulator for the Amiga. It's supposedly a combination of hardware and software; you have to buy the Mega II chip from an Apple dealer to make it work.

Commodore may offer this emulator themselves, in the same way that IBM has.

The Bandito hears that this is being looked at very seriously inside the company. After all, the A500 is the perfect computer for students. With the new low pricing, it would be very attractive to schools if they could run their old Apple software. And Commodore would have a tremendous PR boost, as well as a great way to get more parents to buy Amigas as home computers. Or even Amiga CD-ROM players.

Speaking of Apple IIs, the Bandito thinks it's very amusing how the rats desert a sinking ship. First it was all the Atari software publishers heading for the Amiga, once it became clear that Atari software sold about as well as Communist Party memberships in East Germany. Now, it's Apple II hardware peripherals manufacturers who have seen their native market dry up and blow away. No problem! They'll just change the labels on the stuff, switch a few resistors, and there they are—Amiga peripherals! We'll see how well they do. There are already some well-established vendors in the Amiga market who have travelled a long, hard road to get where they are now, and they'll fight hard to keep their market share. To give them credit, the Bandito has noted some interesting ideas that the new kids are working on; among them is a fax modem for the Amiga. You should be able to buy one by the fall. Of course, by then you'll be seeing fax machines for $300.

OTHER NEWS

M.A.S.T. still hasn't released their Flick-Off board; supposedly, it will ship within the next month. They apparently have a deal with Hitatchi to manufacture a miniaturized VLSI version of the Flick-Off which plugs into the Denise socket. They are also claiming complete compatibility with the ECS under WorkBench 2.0.

Commodore 1-year warranties? Yes, it's real. Commodore decided to get on the bandwagon and offer a real warranty for their products. It doesn't really cost them that much extra, since if a computer is going to die it usually does so when you plug it in for the first time or shortly thereafter. But it's a nice gesture, anyway.

SOFTWARE WARS, CONTINUED...

Looks like the word processing wars are heating up, with the new version of ProWrite going head to head with Pen Pal. Once again, battling software publishers help the buyers by providing more features. Which will be the first to take full advantage of the new super resolution modes offered by the ECS? Neither publisher is talking, but they're both hard at work. The Bandito would like to work in 1280 by 400 for those extra-long sentences.

An update from the front in the HAM Paint Wars. The battlefield is quiet now, littered with the corpses of failed contenders. Photon Paint is no longer being advertised, and DeluxePhotolab never was. Digi-Paint 3 has become the favorite. The latest wrinkle is Digi-Mate 3 by MindWare, which adds animation capabilities to Digi-Paint 3 by using the ARexx connection. If you've got enough memory, run Digi-View at the same time so that you can pull an image in, paint on it, then animate it. More RAM, anybody? Next up: a 24-bit version of Digi-Paint, with a host of new features that can be used in any mode. At least that's what is in the laboratories; there's no word on vhen it might make it to the marketplace.

Finally, there's some conflict and some consolidation going on in the Amiga magazine market. Several names have disappeared, and the Bandito hears they may reappear later by combining operations. We'll see.

Meanwhile, tune in here for the latest industry news and gossip. Has anyone noticed how some of the other magazines are now running "rumors" columns by trying to steal the Bandito's material?

Nice try, but no panatela, fellas. Read the latest news here. Accept no imitators.

THIS GAME IS WAR DEPT.

Nintendo's hot-selling Gameboy portable video game gadget is going to get more competition than the Lynx. Sega Enterprises is producing the Gamegear (with a color LCD screen) for release in Japan possibly this September. It's similar in size to the Lynx, and the specs are interesting. The Gamegear uses an 8-bit Z80A CPU, and displays 32 colors from a palette of 4,096 on its 3.2-inch LCD screen. Sound familiar?

Meanwhile, over 2 million Gameboys have already sold in Japan with almost 2 million overseas. Total sales at home and abroad should hit ten million units by the end of the year.

And NEC is heading to the marketplace with a color LCD portable; their gimmick is that the unit uses the same cartridges as their TurboGrafx machine. Looks like a highly competitive place to be. Say, Commodore, how about a handheld Amiga? It could be done if you put the chips into CMOS. Now there would be the basis for a laptop...

THIS AMIGA LOOKS LIKE AN ATARI ST DEPT.

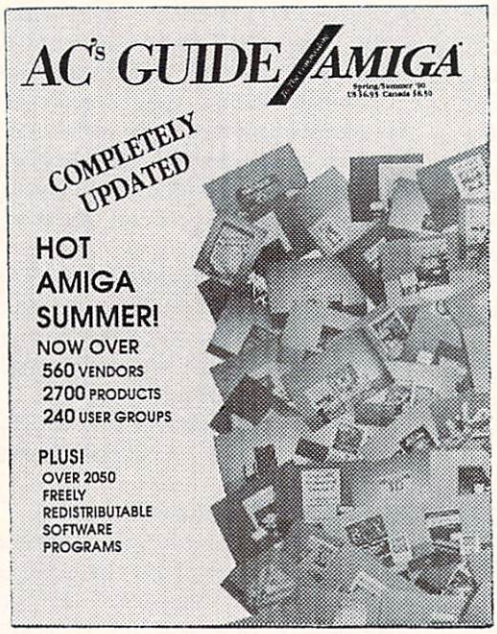
An Atari ST emulator for the Amiga? Why, you ask? Who knows, but there is one or two floating around. One of them is supposedly PD, created by some hackers in Australia, but it is also said to contain some Atari code that is not in the public domain. And it doesn't work very well, from what the Bandito hears. A more interesting product is said to be in development, combining hardware and software in a manner very similar to the successful A-Max emulator. Whether anybody ever ships this product is a matter for another discussion. Before you laugh too hard, consider that such an emulator might help persuade a die-hard Atari fan to switch to the Amiga, because he could use much of his old ST software until he had the bucks to buy Amiga software. Yeah, it sounds cheap, but then only tightwads bought STs in the first place.

In other Atari news, Atari Taiwan was cited for pirating MS-DOS business software. Atari says it was individual employees, because all of their loyal employees use Atari STs in the office. Boy, the hardships you have to go through to work for Atari. By the way, Atari's latest sales figures show that sales and profits were both down for the last quarter; sales were only $85 million (and still dropping). Say, maybe they should merge with Mediagenic...

•AC•

It's Hip!
It's Hot!
It's Happening!

It's Here!



COMPLETELY UPDATED

- Over **2700** Products
- More than **560** Vendors
- More than **2050** Freely Redistributable programs

Now at your local Amazing Dealer.

One From Column A,
Three From Column B.
Using Amiga Menus, In C

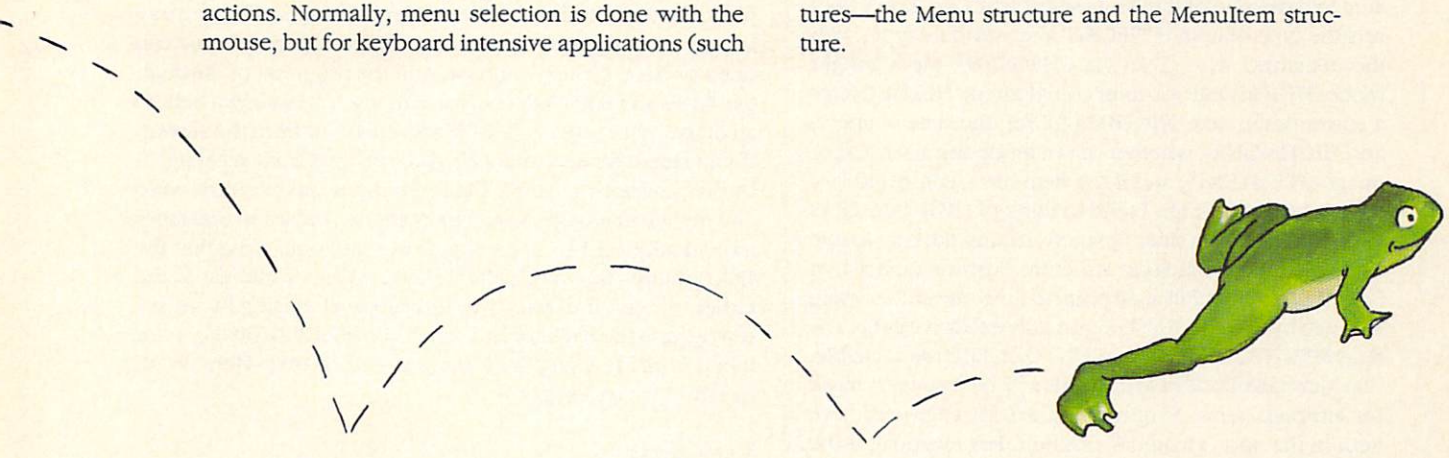
Crunchy Frog II

by Jim Fiore

IN OUR LAST EPISODE, WE LEFT HELEN AS SHE WAS ABOUT TO BE CAPTURED by the giant, ferocious, gorilla from the planet Klien-Horst and—oops, sorry, wrong article. Anyway, last time (AC V4.2) we looked at the ground floor construction of an Amiga C program.

As you may recall we opened a few libraries, a screen, a window, and responded to IDCMP events. This included items such as updating the mouse coordinates and ending the program. In this article we are going to expand on the original program. Our prime item of interest here is the addition of menus. We are also going to open a second window and look at one way of differentiating its IDCMP events from those of other windows. A few little odds and ends will be thrown in as well.

Anyone who has used Workbench knows what a menu is. Simply put, it allows for a convenient way of having the user set program attributes or initiate actions. Normally, menu selection is done with the mouse, but for keyboard intensive applications (such as a word processor), command key 'short-cuts' may be preferable. On the Amiga, each window is allowed to have its own set of menus. Usually a window will have a number of menus, each containing several menu items. These menu items may even have their own set of sub-menu items. No matter how many menus, items, or sub-items exist in a program, only two different types of items/sub-items appear. Basically, you can do one of two things with a menu; either set an ATTRIBUTE (as in the color of your word processor's cursor), or initiate an ACTION (as in starting the spelling checker function). In order to setup menus, we must investigate two important Intuition structures—the Menu structure and the MenuItem structure.



```
struct Menu
{
   struct Menu *NextMenu;
   SHORT LeftEdge, TopEdge, Width, Height;
   USHORT Flags;
   BYTE *MenuName;
   struct MenuItem *FirstItem;
}
```

The NextMenu field of the Menu structure will allow us to make a linked list of menus for our windows. In this manner, we can have several menus for a given window. The last Menu in the list must have this field set to NULL (0 long). The next four fields determine placement along the windows title bar. Presently, TopEdge and Height are ignored, and info from the associated screen is used instead. The Flags field is shared between you and Intuition. The Flag MIDRAWN indicates that this menu is presently being displayed. The Flag MENUENABLED determines whether or not this menu is currently enabled. If menus are not enabled, the associated items will be ghosted, and thus the user will be unable to access them. It is possible to enable/disable menus via the OnMenu() and OffMenu() functions. MenuName is a pointer to a null terminated string, which will appear in the title bar at the position marked by LeftEdge. The first item below this title is linked in via the FirstItem field. The structure for menu items follows:

```
struct MenuItem
{
   struct MenuItem *NextItem;
   SHORT LeftEdge, TopEdge, Width, Height;
   USHORT Flags;
   LONG MutualExclude;
   APTR ItemFill;
   APTR SelectFill;
   BYTE Command;
   struct MenuItem *SubItem;
   USHORT NextSelect;
}
```

As is typical in Intuition structures, the first field allows a list of items to be strung together. As usual, the last item in the list must be set to NULL. The next four fields set the position of the item inside of the larger menu area and the size of the items select box. The Flag's field sets the type of item (CHECKIT if an attribute type, with the initialized state given via CHECKED), the highlight mode (HIGHCOMP for color complement, HIGHBOX for a surrounding box, HIGHIMAGE for alternate imagery, and HIGHNONE), whether or not rendering is via text or image (ITEMTEXT), and if the item has a command key shortcut (COMMSEQ). I tend to think of HIGHIMAGE as HIGHALTERNATE, since 'imagery' means either an Image structure OR an IntuiText structure (just remember that SelectFill and ItemFill must point to the same kind of data, as noted by ITEMTEXT). For mutually exclusive items, the flags MENUTOGGLE and MENUTOGGLED are available. The MutualExclude field contains a 32 bit exclusion mask for attribute items. Simply place a 1 for each exclusive item in the appropriate bit position. For example, if the first three items are mutually exclusive, then the first item will have this field set to binary 00000000000000000000000000000110 (ie., 0x6). If there is no mutual exclusion, this value is zero. ItemFill points to data used to render this item. Normally, this field points to an IntuiText structure, although Image structures are possible. The SelectFill field points to data used if the HIGHIMAGE form of highlighting is specified. Again, it is possible to use alternate IntuiText as well as alternate Image here. (A word of caution: if you decide to use alternate text, make sure that the IntuiText draw mode is set to JAM2 and that the two strings are the same size. If this is not the case, you will end up with character 'overstrike' problems). If the COMMSEQ flag is set, the Command field should hold the character desired. This character should be declared inside single quotes, as in 'A'. The legal characters include all letters, numerals, and elements such as the comma, period, semi-colon, etc. If an item has sub items, the next field will point to an appropriate MenuItem. Note that there is no such thing as a MenuSubItem structure, and that sub-items cannot have sub-sub-items (this field is ignored).

The final field is NextSelect. This field is used by Intuition for the purpose of allowing 'extended selection'. Whenever a menu event is received, this field should be examined to see if there are further menu choices. If this is the last item chosen, this field will be set to MENUNULL. If you would like to fiddle around with the text, here is the IntuiText structure declaration.

```
struct IntuiText
{
 UBYTE FrontPen, BackPen;      /* character and background colors */
 UBYTE DrawMode;               /* JAM1 or JAM2 */
 SHORT LeftEdge, TopEdge;
 struct TextAttr *ITextFont;
 UBYTE *IText;                 /* NULL terminated string */
 struct IntuiText *NextText;
}
```

Briefly, the new program is going to open a second window (text_wind), attaching menus to this window and the original window. You will notice that the declarations for the menus are in three parts. The first part consists of the IntuiText declarations which specify the character string used, a relative position, and the text color. The second part is the listing of MenuItems. I declare one large array since I'm not fond of giving every menu item its own name (besides, this way I get to declare everything in logical order). The third part is the declaration of the Menu. The main window has a single menu, while the text window has three menus (once again, an array is used). Our main window menu choices are simple: we can open or close the text window; quit the program; or, through the 'Odds and Ends' sub choices, either send the screen behind all of the other screens (like Workbench), or flash the screen. If you select 'Screen to Back', you can set it front-most again by pressing left-Amiga-M. The text window menus let us write one of three messages to the text window, erase the messages individually, and set the color of the message. Note that the Color menu uses ATTRIBUTE items. Also of interest is the rather 'cheesy' (but effective) technique of erasing by simply rewriting in the background color. The final item on the Color menu is made using an image instead of text. Here is the structure for an Image:

```
struct Image
{
 SHORT LeftEdge, TopEdge;
 SHORT Width, Height, Depth;
 SHORT *ImageData;
 UBYTE PlanePick, PlaneOnOff;
 struct Image *NextImage;
}
```

➤

AC Disks

Source code and executable programs included for all articles printed in *Amazing Computing*.

AC V3.8 and AC V3.9

Gels In MultiForth Parts I & II: Learn how to use Gels in MultiForth. Author: John Bushakra

FFP & IEEE: An Example of using FFP & IEEE math routines in Modula-2. Author: Steve Faiwiszewski

CAI: A complete Computer Aided Instruction program with editor written in AmigaBASIC. Author: Paul Castonguay

Tumblin' Tots: A complete game written in Assembly language. Save the falling babies in this game. Author: Davd Ashley

VGad: A gadget editor that allows you to easily create gadgets. The program then generates C code that you can use in your own programs. Author: Stephen Vermeulen

MenuEd: A menu editor that allows you to easily create menus. The program then generates C code that you can use in your own programs. Author David Pehrson

Bspread: A powerful spread sheet program written in AmigaBASIC. Author Bryan Cately

AC V4.3 and AC V4.4

Fractals Part I: An introduction to the basics of fractals with examples in AmigaBASIC, True BASIC, and C. Author: Paul Castonguay

Shared Libraries: C source and executable code that shows the use of shared libraries. Author: John Baez

MultiSort: Sorting and intertask communication in Modula-2. Author: Steve Faiwiszewski

Double Playfield: Shows how to use dual playfields in AmigaBASIC. Author: Robert D'Asto

'881 Math Part I: Programming the 68881 math coprocessor chip in C Author: Read Predmore

Args: Passing arguments to an AmigaBASIC program from the CLI. Author: Brian Zupke

AC V4.5 and AC V4.6

Digitized Sound: Using the Audio.device to play digitized sounds in Modula-2. Author: Len A. White

'881 Math Part II: Part II of programming the 68881 math coprocessor chip using a fractal sample. Author: Read Predmore

At Your Request: Using the system-supplied requestors from AmigaBASIC. Author: John F. Weiderhirn

Insta Sound: Tapping the Amiga's sound from AmigaBASIC using the Wave command. Author: Greg Stringfellow

MIDI Out: A MIDI program that you can expand upon. Written in C. Author: Br. Seraphim Winslow

Diskless Compiler: Setting up a compiler environment that doesn't need floppies. Author: Chuck Raudonis

AC V4.7 and AC V4.8

Fractals Part II: Part II on fractals and graphics on the Amiga in AmigaBASIC and True BASIC. Author: Paul Castonguay

Analog Joysticks: The code for using analog joysticks on the Amiga. Written in C. Author: David Kinzer

C Notes: A small program to search a file for a specific string in C. Author: Stephen Kemp

Better String Gadgets: How to tap the power of string gadgets in C. Author: John Bushakra

On Your Alert: Using the system's alerts from AmigaBASIC. Author: John F. Wiederhirn

Batch Files: Executing batch files from AmigaBASIC. Author: Mark Aydellotte

C Notes: The beginning of a utility program in C. Author: Stephen Kemp

AC V4.9

Memory Squares: Test your memory with this AmigaBASIC game. Author: Mike Morrison

High Octane Colors: Use dithering in AmigaBASIC to get the appearance of many more colors. Author: Robert D'Asto

Cell Animation: Using cell animation in Modula-2. Author: Nicholas Cirasella

Improving Graphics: Improve the way your program looks no matter what screen it opens on. In C. Authro: Richard Martin

Gels in Multi-Forth-Part 3: The third and final part on using Gels in Forth. Author: John Bushakra

C Notes V4.9: Look at a simple utility program in C. Author: Stephen Kemp

1D_Cells: A program that simulates a one-dimensional cellular automata. Author: Russell Wallace

Colourscope: A shareware program that shows different graphic designs. Author: Russell Wallace

ShowILBM: A program that displays lo-res, hi-res, interlace and HAM IFF pictures. Author: Russell Wallace

Labyrinth_II: Roll playing text adventure game. Author: Russell Wallace

Most: Text file reader that will display one or more files. The program will automatically format the text for you. Author: Russell Wallace

Terminator: A virus protection program. Author: Russell Wallace

AC V4.10 & AC V4.11

Typing Tutor: A program written in AmigaBASIC that will help you improve your typing. Author: Mike Morrison

Glatt's Gadgets: Using gadgets in Assembly language. Author: Jeff Glatt

Function Evaluator: A program that accepts mathamatical functions and evaluates them. Written in C. Author: Randy Finch

Fractals: Part III: AmigaBASIC code that shows you how to save/load pictures to disk. Author: Paul Castonguay

More Requestors: Using system calls in AmigaBASIC to build requestors. Author: John Wiederhirn

Multi-Forth: Implementing the ARP library from Forth. Author: Lonnie A. Watson

Search Utility: A file search utility written in C. Author: Stephen Kemp

Fast Pics: Re-writing the pixel drawing routine in Assembly language for speed. Author: Scott Steinman

64 Colors: Using extra-half-brite mode in AmigaBASIC. Author: Bryan Catley

Fast Fractals: A fast fractal program written in C with Assembly language subroutines. Author: Hugo M. H. Lyppens

Multitasking in Fortran: All the hard work is done here so you can multitask in Fortran. Author: Jim Locker

AC V4.12 & AC V5.1

Arexx Part II: Information on how to set up your own ARexx programs with examples. Author: Steve Gilmor.

Leggo My LOGO: A Logo program that generates a Christmas tree with decorations. Author: Mike Morrison.

Trees and Recursion: An introduction to binary trees and how to use recursion. Written in C. Author: Forest Arnold.

C Notes: A look at two data compressing techniques in C. Author: Stephen Kemp.

Animation? BASICally: Using cell animation with AmigaBASIC. Author: Mike Morrison

Menu Builder: A utility to help build menus in your own programs. Written in C. Author: Tony Preston.

Dual Demo: How to use dual playfields to make your own arcade games. Written in C. Author: Thomas Eshelman.

Scanning the Screen: Part four in the fractals series. This article covers drawing to the screen. In AmigaBASIC and TrueBasic. Author: Paul Castonguay.

C Notes: Recursive functions in C. Author: Stephen Kemp.

AC V5.2. & 5.3

Dynamic Memory!: Flexible string gadget requester using dynamic memory allocation. Author: Randy Finch.

Call Assembly language from BASIC: Add speed to your programs with Assembly. Author: Martin F. Combs.

Conundrum: An AmigaBASIC program that is a puzzle-like game, similar to the game Simon. Author: Dave Senger.

Music Titler: Generates a titler display to accompany the audio on a VCR recording. Author Brian Zupke

C Notes From the C Group: Writing functions that accept a variable number of arguments. Author: Stephen Kemp

Screen Saver: A quick remedy to prolong the life of your monitor. Author: Bryan Catley

AC V 5.4 & AC 5.5

Bridging The 3.5" Chasm: Making Amiga 3.5" drives compatible with IBM 3.5" drives. Author: Karl D. Belsom.

Ham Bone: A neat program that illustrates programming in HAM mode. Author: Robert D'Asto.

Handling Gadget and Mouse IntuiEvents: More gadgets in Assembly language. Author: Jeff Glatt.

Super Bitmaps in BASIC: Holding a graphics display larger than the moitor screen. Author: Jason Cahill

Rounding Off Your Numbers: Programming routines to make rounding you numbers a little easier. Author: Sedgwick Simons

Mouse Gadgets: Faster BASIC mouse input. Author: Miichael Fahrion

Print Utility: A homemad print utility, with some extra added features. Author: Brian Zupke

Bio-feedback/Lie detector Device: Build your own lie detector device. Author John Iovine.

Do It By Remote: Build an Amiga-operated remote controller for your home. Author: Andre Theberge

AC V5.6 & V5.7

Convergence: Part five of the Fractal series. Author Paul Castonguay

Amiga Turtle Graphics: Computer graphics and programming with a LOGO-like graphics system. Author Dylan MnNamee

C Notes: Doing linked list and doubly linked lists in C. Author Stephen Kemp

Tree Traversal & Tree Search: Two common methods for traversing trees. Author: Forest W. Arnold

Exceptional Conduct: A quick response to user requests, achieved through efficient program logic. Author: Mark Cashman.

Getting to the Point: Custom Intuition pointers in AmigaBASIC. Author: Robert D'Asto

Crunchy Frog II: Adding windows and other odds and ends. Author: Jim Fiore

Synchroncitity: Right and left brain lateralization. Author: John Iovine

C Notes From the C Group: Doubly linked lists revisited. Author: Stephen Kemp

Poor Man's Spreadsheet: A simple spreadsheet program that demonstrates manipulating arrays. Author: Gerry L. Penrose.

For PDS orders, please use form on page 96
Visa and MasterCard is available on orders of $20.00 or more.

The Width, Height, and Depth fields describe the layout of the image data pointed to by the ImageData field. If you use fewer bitplanes than the screen allows, you can decide where to place your bitplanes with the PlanePick field. The remaining bitplanes are turned on or off according to to the PlaneOnOff field. Our image is a single bit plane, so please feel free to experiment with the PlanePick and PlaneOnOff fields. Also, when creating image data, you will normally do so with a 'tool' that looks like a paint program, but which outputs C source directly. These tools are available commercially and in the public domain.

When using menus, you have to inform Intuition that a particular menu should be attached to a given window through the SetMenuStrip() function. The complimentary function is called ClearMenuStrip() and is called in damp_mop(). It is possible to attach different menus to a window by Clearing and Setting menu strips. Note that both windows and their menus are created and displayed when the program is first run.

Since we now have two windows opened, we must deal with IDCMP events from two sources. There are different ways of doing this. In this program we simply Wait() on signals from either window. Note that it is necessary to test for the existence of the text window, since it is possible to close it and still have the program remain active. When the program is signaled, we compare the wait_mask to the available wait bits to determine which window was used. At this point we branch to one of two IDCMP routines. Note that the new handle_text_messages() routine is basically a copy of the old handle_main_messages() function.

In order to process the IDCMP MENUPICK messages, we have two functions; handle_text_menu() and handle_main_menu(). Both of these functions do essentially the same thing; that is, they break down the menu code variable and respond accordingly. Note that each handle_menu routine contains a large while() loop. This is used to capture 'extended select' menu items (the next item is obtained by examining the NextSelect field of the MenuItem returned from the call to ItemAddress()). Inside the loop, the macro MENUNUM() returns the ordinal value of our selected menu. ITEMNUM() is used to find the ordinal value of the chosen menu item. If this item has sub items, their ordinal value is obtained from the macro SUBNUM(). In this way we can 'track down' any menu choice. Note that the first menu, item, or sub item actually holds position 0 and not position 1 (just like everything else in C, numbering starts at 0!).

There are two remaining items in the program that I would like to draw to your attention. First, you will notice that our old dummy handler handle_main_newsize() has been modified. In working with the old program, you may have noticed that shrinking the main window destroys the sine wave drawing. It would be nice if we could redisplay the wave when the window is increased in size again. That is exactly what the call to setup_main_wind() does. If you shrink the window and then expand it, you will notice that the wave is redrawn. Admittedly, this is very crude, but it does show how NEWSIZE events can be used. A better scheme would examine the size and see if redrawing is required (after all, you could make the window larger than its initial size with no destruction of the graph). With particularly complex renderings you might even choose to do your own image backup and copying. The second item of interest is the function close_text_window(). Note that this function can be called from three places: 1) user selects the text windows Close gadget, 2) user selects 'Close Text Window' from the main menu, and 3) user kills program either from the main menu or the main windows Close gadget (both call damp_mop(), which calls close_text_window()). The first thing you will notice in this function is that the text windows IDCMP que is drained of any 'left over' messages (such as MOUSEMOVEs). Next, the menu strip is cleared and the window closed. Finally, the test_wind pointer is NULLed. This last line is VERY important. If we do this consistently, we can effectively use this pointer as a 'window exists' flag. In other words, if the value of the pointer is zero, we know that the window is not open, and therefore we cannot render to it or expect to receive IDCMP messages from it. On the other hand, if the pointer is non-null, we can assume that the window is open and available. You will notice that this technique is used throughout this program. In essence, statements such as if(text_wind) are really shorthand for saying 'If this window is presently open, then...'. Examples are found in the main() FOREVER loop and in handle_main_messages() (in case the user tries to open the text window and it is already open).

Well, that about wraps it up for this installment. At this point you now have exposure to some of the important base elements of an Amiga program. There are a number of interesting topics yet to be explored, though, so stay tuned!

Author's info:

Jim Fiore is the resident C programmer at dissidents in Utica, NY. He has a number of years of teaching experience in Electrical Engineering Technology as well. In his spare time Jim enjoys falling off of things, and listening to Frank Zappa, King Crimson, and Kate Bush. He may be contacted through BIX as jfiore.

Listing

```
/*                          CrunchyFrog2.c

   Jim Fiore @ dissidents 10/9/88. Updated 8/22/89. This program is
   copyrighted, however, you can use it, with the exception of
distribution for a profit.

   compiled and linked with Manx Aztec C v3.6 under AmigaDOS 1.3

   cc +L CrunchyFrog2.c

   ln +cdb CrunchyFrog2.o -lm32 -lc32

   Image data must be in CHIP RAM.( +cdb for Manx is the easy way out.)

*/

#include "functions.h"
#include "intuition/intuition.h"
#include "math.h"

/* ————————— defines ————————— */

/* MyText(*RastPort, Xposition, Yposition, char *buffer) */

#define MyText(r,x,y,b)  Move((r),(x),(y)); Text((r),(b),strlen(b))


#define INTUITION_REV 33L
#define GRAPHICS_REV  33L
#define DEPTH         3
#define MAX_COLORS    8   /* 2 raised to DEPTH */

/* ————————— Globals ————————— */

struct IntuitionBase *IntuitionBase=0L;
struct GfxBase       *GfxBase=0L;
struct Window        *main_wind=0L;
struct Screen        *main_scrn=0L;
struct ViewPort      *view_port=0L;


/* use the 80 character topaz font for the screen lettering */

struct TextAttr topaz80_font_attr={
 (UBYTE *)"topaz.font",
 TOPAZ_EIGHTY,
 FS_NORMAL,
 FPF_ROMFONT };


struct NewScreen ns={
 0, 0,                              /* LeftEdge, TopEdge */
 640, 200,                          /* Width, Height */
 DEPTH, 0, 1,                       /* Depth. DetailPen, BlockPen */
 HIRES,                             /* ViewModes */
 SCREENBEHIND | CUSTOMSCREEN,       /* Type */
 &topaz80_font_attr,                /* Font */
 (UBYTE *)" Dinsdale's Screen ",    /* DefaultTitle */
 NULL, NULL };                      /* Gadgets, CustomBitMap */


struct NewWindow m_nw={
 50, 20,                            /* LeftEdge, TopEdge */
 300, 120,                          /* Width, Height */
 -1, -1,                            /* DetailPen, BlockPen (default) */
 MENUPICK | GADGETUP | GADGETDOWN | \
 RAWKEY | CLOSEWINDOW | MOUSEMOVE | \
 MOUSEBUTTONS | NEWSIZE,                 /* IDCMPFlags */
 SMART_REFRESH | ACTIVATE | \
 WINDOWSIZING | WINDOWCLOSE | \
 WINDOWDEPTH | WINDOWDRAG | \
 REPORTMOUSE,                       /* Flags */
 NULL, NULL,                        /* FirstGadget, CheckMark */
 (UBYTE *)" Main Window ",          /* Title */
 NULL, NULL,                        /* Screen, BitMap */
 100, 50,                           /* MinWidth, MinHeight */
 640, 200,                          /* MaxWidth, MaxHeight */
 CUSTOMSCREEN };                    /* Type */


/* data declarations which are new for 2 */

UBYTE msg1[70]={"Some say that Heaven is Hell"};
UBYTE msg2[70]={"Some say that Hell is Heaven"};
UBYTE msg3[70]={"Some say Ha Ha Ha"};


struct Window        *text_wind=0L;

struct NewWindow txt_nw={
 250, 50,                           /* LeftEdge, TopEdge */
 250, 100,                          /* Width, Height */
 -1, -1,                            /* DetailPen, BlockPen (default) */
 MENUPICK | GADGETUP | GADGETDOWN | \
 RAWKEY | CLOSEWINDOW | MOUSEMOVE | \
 MOUSEBUTTONS | NEWSIZE,                 /* IDCMPFlags */
 SMART_REFRESH | ACTIVATE | \
 WINDOWSIZING | WINDOWCLOSE | \
 WINDOWDEPTH | WINDOWDRAG | \
 REPORTMOUSE,                       /* Flags */
 NULL, NULL,                        /* FirstGadget, CheckMark */
 (UBYTE *)" Text Window ",          /* Title */
 NULL, NULL,                        /* Screen, BitMap */
 100, 50,                           /* MinWidth, MinHeight */
 640, 200,                          /* MaxWidth, MaxHeight */
 CUSTOMSCREEN };                    /* Type */



/* main menu segment */

struct IntuiText main_itxt[]={
 {0,1,JAM1,3,1,NULL,(UBYTE *)"Open Text Window"},
 {0,1,JAM1,3,1,NULL,(UBYTE *)"Close Text Window"},
 {0,1,JAM1,3,1,NULL,(UBYTE *)"Odds and Ends"},
 {0,1,JAM1,3,1,NULL,(UBYTE *)"Flash Screen"},
 {0,1,JAM1,3,1,NULL,(UBYTE *)"Screen to Back"},
 {0,1,JAM1,3,1,NULL,(UBYTE *)"-Quit-"},};


struct MenuItem main_mi[]={
 {&main_mi[1],0,0,144,10,
  (ITEMTEXT  | ITEMENABLED | HIGHCOMP),
  NULL,(APTR)&main_itxt[0],NULL,NULL},

 {&main_mi[2],0,10,144,10,
  (ITEMTEXT | ITEMENABLED | HIGHCOMP),
  NULL,(APTR)&main_itxt[1],NULL,NULL},

 {&main_mi[5],0,20,144,10,
  (ITEMTEXT | ITEMENABLED | HIGHCOMP),
  NULL,(APTR)&main_itxt[2],NULL,NULL,&main_mi[3]},
```

```
    {&main_mi[4],120,0,130,10,
     (ITEMTEXT | ITEMENABLED | HIGHCOMP),
     NULL,(APTR)&main_itxt[3],NULL,NULL},

    {NULL,120,10,130,10,
     (ITEMTEXT | ITEMENABLED | HIGHCOMP),
     NULL,(APTR)&main_itxt[4],NULL,NULL},

  {NULL,0,30,144,10,
   (ITEMTEXT | ITEMENABLED | HIGHCOMP),
   NULL,(APTR)&main_itxt[5],NULL,NULL},};

struct Menu main_menu ={NULL,0,0,72,0,MENUENABLED,  "Project
",&main_mi[0]};



/* text_wind segment */


USHORT image_data[]=
/* 48 wide by 20 high by 1 plane deep */
/* plain number 0 */

{ 0x3ff0, 0x0000, 0x0000,
  0x601f, 0xf000, 0x0000,
  0x3800, 0x1f00, 0x0000,
  0x0780, 0x01e0, 0x0000,
  0x007e, 0x003f, 0x0000,
  0x0003, 0xe1c0, 0xfe00,
  0x0000, 0x3700, 0x03e7,
  0x007f, 0xdc00, 0x003e,
  0x00c0, 0x3001, 0x0002,
  0x00c0, 0x6007, 0x0002,
  0x0078, 0x701d, 0xc002,
  0x00cf, 0xfc28, 0x3fe2,
  0x00c0, 0x3ff8, 0x0006,
  0x0078, 0x0020, 0x0004,
  0x003f, 0xffe0, 0x0004,
  0x0060, 0x0040, 0x0004,
  0x0070, 0x00c0, 0x0034,
  0x000f, 0xffff, 0x83ec,
  0x0000, 0x0000, 0xfc7f,
  0x0000, 0x0000, 0x0000 };

struct Image menu_image = { 0,0, 48,20,1, image_data, 0x2, 0x0, NULL };

struct IntuiText text_itxt[]={
  {0,1,JAM1,2,1,NULL,(UBYTE *)"Message 1"},
  {0,1,JAM1,2,1,NULL,(UBYTE *)"Message 2"},
  {0,1,JAM1,2,1,NULL,(UBYTE *)"Message 3"},
  {0,1,JAM1,2,1,NULL,(UBYTE *)"Message 1"},
  {0,1,JAM1,2,1,NULL,(UBYTE *)"Message 2"},
  {0,1,JAM1,2,1,NULL,(UBYTE *)"Message 3"},
  {0,1,JAM1,2+CHECKWIDTH,1,NULL,(UBYTE *)"Color 1"},
  {2,1,JAM1,2+CHECKWIDTH,1,NULL,(UBYTE *)"Color 2"},
  {3,1,JAM1,2+CHECKWIDTH,1,NULL,(UBYTE *)"Color 3"},};



/* Draw */

struct MenuItem text_mi[]={
  {&text_mi[1],0,0,(90+COMMWIDTH),10,
   (ITEMTEXT  | ITEMENABLED | HIGHCOMP | COMMSEQ),
   NULL,(APTR)&text_itxt[0],NULL,'1'},

  {&text_mi[2],0,10,(90+COMMWIDTH),10,
   (ITEMTEXT | ITEMENABLED | HIGHCOMP | COMMSEQ),
   NULL,(APTR)&text_itxt[1],NULL,'2'},

  {NULL,0,20,(90+COMMWIDTH),10,
   (ITEMTEXT | ITEMENABLED | HIGHCOMP | COMMSEQ),
   NULL,(APTR)&text_itxt[2],NULL,'3'},

/* Erase */

  {&text_mi[4],0,0,(90+COMMWIDTH),10,
   (ITEMTEXT | ITEMENABLED | HIGHCOMP | COMMSEQ),
   NULL,(APTR)&text_itxt[3],NULL,'4'},

  {&text_mi[5],0,10,(90+COMMWIDTH),10,
   (ITEMTEXT | ITEMENABLED | HIGHCOMP | COMMSEQ),
   NULL,(APTR)&text_itxt[4],NULL,'5'},

  {NULL,0,20,(90+COMMWIDTH),10,
   (ITEMTEXT | ITEMENABLED | HIGHCOMP | COMMSEQ),
   NULL,(APTR)&text_itxt[5],NULL,'6'},

/* Drawing color */


  {&text_mi[7],0,0,(75+CHECKWIDTH+COMMWIDTH),10,
   (ITEMTEXT | ITEMENABLED | HIGHCOMP | COMMSEQ | CHECKED | CHECKIT |
MENUTOGGLE),
   0x6,(APTR)&text_itxt[6],NULL,'7',NULL},

  {&text_mi[8],0,10,(75+CHECKWIDTH+COMMWIDTH),10,
   (ITEMTEXT | ITEMENABLED | HIGHCOMP | COMMSEQ | CHECKIT | MENUTOGGLE),
   0x5,(APTR)&text_itxt[7],NULL,'8',NULL},

  {&text_mi[9],0,20,(75+CHECKWIDTH+COMMWIDTH),10,
   (ITEMTEXT | ITEMENABLED | HIGHCOMP | COMMSEQ | CHECKIT | MENUTOGGLE),
   0x3,(APTR)&text_itxt[8],NULL,'9',NULL},

 {NULL,10,30,48,20,
  ( ITEMENABLED | HIGHBOX ),
  NULL,(APTR)&menu_image,NULL,NULL,NULL},};



struct Menu text_menu[]={
 {&text_menu[1],0,0,48,0,MENUENABLED,   "Draw   ",&text_mi[0]},
 {&text_menu[2],50,0,56,0,MENUENABLED,   "Erase   ",&text_mi[3]},
 {NULL,110,0,56,0,MENUENABLED,   "Color   ",&text_mi[6]},};


/* done with data declarations, new for 2 */



USHORT sys_color_table[MAX_COLORS]={ 0xabc, 0x130, 0xf00, 0xaa0, 0xbf0,
                                     0x54f, 0xb0e, 0x3aa};


VOID open_all(), damp_mop(), handle_main_messages(), setup_main_wind(),
      handle_main_rawkey(), handle_main_menu(), handle_main_gadgetdown(),
      handle_main_gadgetup(), handle_main_mousebuttons(),
      handle_main_newsize();

/* new for 2 */
VOID handle_text_messages(), handle_text_menu(), close_text_window();



/*----------------start of main()----------------*/

/* this function altered for 2 */
main()

{
 LONG main_wait_bit, wait_mask, text_wait_bit /* <- new for 2 */;


 /* - open Intuition and Graphics libs -- */

 open_all();


 if ( (main_scrn = (struct Screen *)OpenScreen(&ns)) == NULL )
damp_mop();

 m_nw.Screen=main_scrn;

 if ( (main_wind = (struct Window *)OpenWindow(&m_nw)) == NULL)
damp_mop();


 /* stuff that is new for 2 */

 SetMenuStrip( main_wind, &main_menu );


 txt_nw.Screen=main_scrn;

 if ( (text_wind = (struct Window *)OpenWindow(&txt_nw)) == NULL)
damp_mop();

 SetMenuStrip( text_wind, text_menu );

 /* end of new for 2 */



 /*  set screen colors to our choices */

 view_port = ViewPortAddress(main_wind);

 LoadRGB4(view_port, sys_color_table, MAX_COLORS);

 ScreenToFront(main_scrn);


 /* do some drawing */

 setup_main_wind();


 /* - set up IDCMP read loop. This loop altered for 2 - */

 FOREVER          /* also known as for(;;) */
 {
    main_wait_bit = 1<<main_wind->UserPort->mp_SigBit; /* get
main_wind's */
/* signal bit */
```

```
    if ( text_wind )
      text_wait_bit = 1<<text_wind->UserPort->mp_SigBit; /* get
text_wind's*/
    else                                              /* signal
bit */
      text_wait_bit = 0L;

    wait_mask = Wait( main_wait_bit | text_wait_bit ); /* go to
sleep til user does something */

    if (wait_mask & main_wait_bit)  /* true if main_wind woke up
*/
     handle_main_messages();

    if (wait_mask & text_wait_bit)  /* true if main_wind woke up
*/
     handle_text_messages();

 }

}          /* end of main() */



/*--------- opens Intuition and Graphics libs ---------*/

VOID open_all()

{

 IntuitionBase=(struct IntuitionBase
*)OpenLibrary("intuition.library",  INTUITION_REV);

 if (IntuitionBase==NULL) damp_mop();


 GfxBase=(struct GfxBase *)OpenLibrary("graphics.library",
GRAPHICS_REV);

 if (GfxBase==NULL)  damp_mop();

}



/*--------- closes windows, screen, Graphics, Intuition ---------*/

/* this function altered for 2 */
VOID damp_mop()

{
 struct IntuiMessage *mes;

 if ( main_wind )
 {        /* Drain the IDCMP. Actually, this isn't really required
as this memory will be reclaimed, but I sleep better if I do it */
   while( mes=(struct IntuiMessage *)GetMsg( main_wind->UserPort )
)
     ReplyMsg( mes );

   ClearMenuStrip( main_wind );              /* <- new for 2 */
   CloseWindow( main_wind );
 }

 if ( text_wind )      close_text_window();   /* <- new for 2 */

 if ( main_scrn )      CloseScreen( main_scrn );
 if ( GfxBase )        CloseLibrary( GfxBase );
 if ( IntuitionBase ) CloseLibrary( IntuitionBase );
 exit (FALSE);

}

VOID setup_main_wind()  /* draws x,y axis and a sine wave */
{
 struct RastPort *rast = main_wind->RPort;
 double x, y;

 /* Draw x, y axis. First set the pen color to pen 4. The origin
will be at point 20, 50, the max swing will be +/- 35 pixels, and
the length will be 200 pixels  */

 SetAPen( rast, 4);

 Move( rast, 20, 15);
 Draw( rast, 20, 85);

 Move( rast, 20, 50);
 Draw( rast, 220, 50);

/* Now for the sine wave. Move to the origin, and to see it more
clearly, change to pen 5 */

 Move( rast, 20, 50);
 SetAPen( rast, 5);

 for( x=1.0; x<200.0; x=x+1.0)
 {
     y = -1.0 * (35.0 * sin( x/10.0 ));

     /* Offset y so it stradles the x axis */
     y = y + 50.0;

     /* Plot line segment. Note the x offset. Cast is important! */
     Draw( rast, (SHORT)x+20, (SHORT)y );
 }

/* Print the mouse position titles (X, Y) next to where the values will
   appear */

 SetAPen( rast, 6);
 MyText( rast, 20, 100, "X");
 MyText( rast, 80, 100, "Y");


}



VOID update_coords( x, y )     /* prints the mouse x y position */
SHORT x,y;
{
   UBYTE             buf[5];
   struct RastPort  *rast = main_wind->RPort;


   sprintf( buf, "%4d", x);
   MyText( rast, 30, 100, buf);

   sprintf( buf, "%4d", y);
   MyText( rast, 90, 100, buf);

}



/* dummy handlers, to be used in the future, follow */

VOID  handle_main_mousebuttons( code )
USHORT code;
{
```

```
}

VOID  handle_main_rawkey( code, qualifier )
USHORT code, qualifier;
{
}

VOID  handle_main_gadgetdown( address )
APTR address;
{
}

VOID  handle_main_gadgetup( address )
APTR address;
{
}


/* this function altered for 2 */
VOID  handle_main_newsize( window_ptr )
struct Window *window_ptr;
{
  setup_main_wind();
}



   /* this function new for 2 */
VOID close_text_window()
{
 struct IntuiMessage *mes;

 while( mes = (struct IntuiMessage *)GetMsg( text_wind->UserPort ) )
   ReplyMsg( mes );

 ClearMenuStrip( text_wind );
 CloseWindow( text_wind );
 text_wind = 0L;
}



   /* this function altered for 2. It used to be a dummy handler */
VOID  handle_main_menu( code )
USHORT code;
{

  while ( code!=MENUNULL )
  {
     switch(MENUNUM( code ))
     {
         case 0:                    /* Project */
           switch(ITEMNUM( code ))
           {
               case 0:            /* open text window */
                 if( text_wind )
                 {                 /* already open! */
                   DisplayBeep( main_scrn );
                 }
                 else              /* open it */
                 {
                   txt_nw.Screen = main_scrn;
                   if ( (text_wind = (struct Window
*)OpenWindow(&txt_nw))\
                        == NULL) break;
                   SetMenuStrip( text_wind, text_menu );
                 }
                 break;

               case 1:             /* close text window */
                 if( text_wind )   /* don't try to close if !exist */
                  close_text_window();
                 break;

               case 2:                          /* odds and ends */
                 switch(SUBNUM( code ))
                 {
                    case 0:                      /* flash screen */
                      DisplayBeep( main_scrn );
                      break;

                    case 1:                       /* screen to back */
                      ScreenToBack( main_scrn );
                      break;
                 }
                 break;

               case 3:             /* quit */
                 damp_mop();
                 break;

           } /* end of switch(ITEMNUM( code )) */
           break;

         default:
           break;

     } /* end of switch(MENUNUM( code )) */

     code = ItemAddress( &main_menu, code )->NextSelect;

   }  /* end of while( code != MENUNULL ) */

}



/* this function new for 2 */
VOID  handle_text_menu( code )
USHORT code;
{
  struct RastPort *rp = text_wind->RPort;
  static USHORT pen = 1;

  while ( code!=MENUNULL )
  {
     switch(MENUNUM( code ))
     {
         case 0:                    /* Draw */
           SetAPen( rp, pen );
           switch(ITEMNUM( code ))
           {
               case 0:
                 MyText( rp, 10, 20, msg1 );
                 break;

               case 1:
                 MyText( rp, 10, 30, msg2 );
                 break;

               case 2:
                 MyText( rp, 10, 40, msg3 );
                 break;
           }
           break;

         case 1:                    /* Erase */
           SetAPen( rp, 0 );
           switch(ITEMNUM( code ))
           {
               case 0:
                 MyText( rp, 10, 20, msg1 );
                 break;

               case 1:
                 MyText( rp, 10, 30, msg2 );
                 break;

               case 2:
                 MyText( rp, 10, 40, msg3 );
                 break;
           }
           break;

         case 2:                    /* Color */
           switch(ITEMNUM( code ))
           {
               case 0:
                 pen = 1;
                 break;

               case 1:
                 pen = 2;
                 break;

               case 2:
                 pen = 3;
                 break;

               case 3:    /* this is our 'do nothing' image choice */
                 break;

           }
           break;

         default:
           break;

     } /* end of switch(MENUNUM( code )) */

     code = ItemAddress( text_menu, code )->NextSelect;

   }  /* end of while( code != MENUNULL ) */

}

/*--------------- IDCMP routine ---------------*/

VOID handle_main_messages()
{
  struct IntuiMessage  *message;
  SHORT                mx, my;
  static SHORT         mouse_moved;
```

```
  mouse_moved = FALSE;

 /* As long as we have messages in the que, make local copies of
appropriate
    data, reply, and then process accordingly */

  while( message=(struct IntuiMessage *)GetMsg( main_wind->UserPort ) )
  {
     ULONG  class = message->Class;
     USHORT code = message->Code;
     USHORT qualifier = message->Qualifier;
     APTR   address = message->IAddress;
     struct Window *window_ptr = message->IDCMPWindow;

     mx = message->MouseX;  /* Declared above. Must be local to entire
func */
     my = message->MouseY;  /* if we use the mouse move collection
technique */

     ReplyMsg( message );
     switch( class )
     {
          case MOUSEMOVE:
            mouse_moved = TRUE;
     /*     update_coords( mx, my ); */ /* <-uncomment this for 'real
time' updates */
            break;

          case MOUSEBUTTONS:
            handle_main_mousebuttons( code );
            break;

          case CLOSEWINDOW:
            damp_mop();
            break;

          case RAWKEY:
            handle_main_rawkey( code, qualifier );
            break;

          case MENUPICK:
            handle_main_menu( code );
            break;

          case GADGETDOWN:
            handle_main_gadgetdown( address );
            break;

          case GADGETUP:
            handle_main_gadgetup( address );
            break;

          case NEWSIZE:
            handle_main_newsize( window_ptr );
            break;

          /* we could add other choices here as well */
     }

  }          /* end of while(message..) */

  if ( mouse_moved ) update_coords( mx, my );

} /* end of handle_main_messages() */



/* this function new for 2 */
VOID handle_text_messages()
{
  struct IntuiMessage  *message;
  SHORT                mx, my;
  static SHORT         mouse_moved;


  mouse_moved = FALSE;

 /* As long as we have messages in the que, make local copies of
appropriate
    data, reply, and then process accordingly */

  while( message=(struct IntuiMessage *)GetMsg( text_wind->UserPort ) )
  {
     ULONG  class = message->Class;
     USHORT code = message->Code;
     USHORT qualifier = message->Qualifier;
     APTR   address = message->IAddress;
     struct Window *window_ptr = message->IDCMPWindow;

     mx = message->MouseX;  /* Declared above. Must be local to entire
func */
     my = message->MouseY;  /* if we use the mouse move collection
technique */

     ReplyMsg( message );
```



```
     switch( class )
     {
          case MOUSEMOVE:
            mouse_moved = TRUE;
            break;

          case MOUSEBUTTONS:
            break;

          case CLOSEWINDOW:
            close_text_window();
            return;
            break;

          case RAWKEY:
            break;

          case MENUPICK:
            handle_text_menu( code );
            break;

          case GADGETDOWN:
            break;

          case GADGETUP:
            break;

          case NEWSIZE:
            break;

          /* we could add other choices here as well */
     }

   }     /* end of while(message..) */

} /* end of handle_text_messages() */

/********************** Dat's all folks....
*****************************/
```

•AC•

(continued from page 49)

Full Metal Planet a secret space mission and you are one of the best pilots in the galaxy. Gamers are challenged to collect as much ore as possible, one of the planet's most desired natural resources. Fighting for their lives, game players must capture all the ore taken by rival companies on the planet and return home. Available in the Fall. $49.95. **Inquiry # 282**

Data East USA, Inc.
1850 Little Orchard Street
San Jose, CA 95125
(408) 286-7074
FAX (408) 286-2071

Electronic Arts

The creators of Populous now bring ***Powermonger***, a game set in the future where you are the leader of a tribe who arrives on uncharted worlds and has a host of choices to make. Do you negotiate, battle, or join forces with the various leaders you meet, each with his own traits. $49.95. **Inquiry # 283**

Electronic Arts & subsidiaries

California Dreams

Add ***Street Rod*** to the ever-growing list of driving games for the Amiga. Set in the 50's, you have the summer to win races and buy better hot rods to take on the king of the streets and win. A data disk with more cars and parts will also be available. $39.95. **Inquiry # 284**

Fans of Tetris-style games will enjoy playing ***Blockout***, which is based on the 60's children's toy the SOMA cube. You rotate different 3-D blocks down a pit for fill an entire level. Slowly but surely the levels rise on your as you fail to fill them. $39.95. **Inquiry # 285**

Tunnels of Armageddon: you go racing through a network of underground tunnels in search of a doomsday device that threatens all mankind. Disarm it and mankind will be granted interstellar travel. $39.95. **Inquiry # 286**

Wings puts you in the cockpit of a WWI fighter as a rookie pilot who just joined an elite squadron. The game spans WWI and includes all of the fighters of WWI. Over 2,000 missions available. August for $49.95. **Inquiry # 287**

Brainblaster, a set of two games in one. *Xenon 2* and *Bombuzal* are both fast-paced arcade action games. $39.95. **Inquiry # 288**

First Byte

Spell-A-Saurus, is a highly graphic- and sound-oriented game that teaches kids to read while having a good time at it. It includes four word games, *AstroDrive, Zug Escape!, Ptera-Ptutor,* and *Spell-A-Saur*. $44.95. **Inquiry # 289**

Interstel

D.R.A.G.O.N. Force lead an elite strike force of 14 soldiers on anti-terrorist attacks around the world. You get to blow up cocaine labs and rescue hostages. $49.95. **Inquiry # 290**

LucasFilms Games

You play a wet-behind-the-ears would-be pirate who wants to be a dashing buccaneer in ***The Secret of Monkey Island***, a comic game that doesn't take the subject too seriously, complete with 3-D graphics and reggae music. Available late this fall for $59.95. **Inquiry # 291**

The unofficial sequel to ***Their Finest Hour, Secret Weapons of the Luftwaffe*** is set near the end of the war, when Germany was experimenting with jet engine fighters. The player can choose from training missions or tours of duty (25 in all), and nine different American aircraft, fighters and bombers. Due out later in the year. $59.95. **Inquiry # 292**

Miles Computing

Become an underwater Rambo with ***Aquanaut***. Your mission is to stop an underwater alien attack force and free a captive city. $39.95. **Inquiry # 293**

New World Computing

For those who don't take nuclear destruction seriously, ***Nuclear War*** is for you. This comical action/strategy game is based on the popular card game has the serious aspect of fending off nuclear attack mixed with oddities like Cattletech and 16-Ton Weights. $49.95 **Inquiry # 294**

Might and Magic II now it comes to the Amiga, complete with 3-D graphics, automapping and a level of play like you've never seen. $59.95 **Inquiry # 295**

Ocean

F29 Retaliator gives you the chance to fly the hot rod fighter with the forward-swept wings. The 99 different scenes allow dogfights, ground strikes, attacking sea-going targets, and more. Due in August. $49.95. **Inquiry # 296**

Horror fans are in for a treat, Clive Barker's ***Nightbreed***. You play Boone, the hero of the story, running from accusations of murder and into a city of shapeshifters. August. $49.95 **Inquiry # 297**

Billy The Kid: one or two players can enjoy this wild west game and assume the role of Billy or the sheriff, Pat Garrett. Historically accurate with over a half hour of MIDI music. Due later this year. $49.95. **Inquiry # 298**

If you liked Carrier Command, then you should be all ready for ***Battle Command***. Created by CC's designers, the game is a futuristic armored tank battle simulator. August.$49.95. **Inquiry # 299**

The Untouchables puts you in Eliot Ness' shoes to take on the Mob, reliving all of the legendary battles Ness and his men had with Capone's hoods. July.$39.95. **Inquiry # 300**

Lost Patrol places you as the commander of a crew of helicopter pilots who are shot down over the Killing Fields and try to get out alive. August. $49.95. **Inquiry # 301**

SSI

The DragonLance series comes to the Amiga one again, this time in the form of ***DragonStrike***. You play a Solamnic Knight of Krynn with a dragon steed. You fly with other good dragons into combat against other dragons, wyverns, ships, archers, and flying citadels. Due this fall $49.95 **Inquiry # 302**

UBI Soft

In the game ***Unreal***, a bizarre twist occurs when the hero of the story befriends a dragon to save a helpless maiden. Due in August for $49.95 **Inquiry # 303**

In ***B.A.T.***, you play a special agent looking for corporate leaders who have been banished from Earth and now want to destroy it with bacterial weapons.Due in early summer. $49.95. **Inquiry # 304**

Electronics Arts
1820 Gateway Dr.
San Mateo, CA 94404
(800) 245-4525
(415) 571-7171
FAX (415) 571-7995

InnerPrise Software, Inc.

World Of Turrican, will feature 13 levels with 1,300 screens, hidden rooms, multi-directional parallax scrolling, and thirty different sound effects. $39.95 **Inquiry # 305**

The Plague, you chase after a lab experiment gone hog wild. With forty-two colors and large-sized sprites, expect a screen full of colors and sights. $39.95 **Inquiry # 306**

Globulous mixes arcade action with adventure-style puzzles. 25 different screens and a 3-D isometric background, and there is a unique "flip screen" feature that inverts the screen during play. $29.95 **Inquiry # 307**

InnerPrise Software, Inc.
128 Cockeysville Road
Hunt Valley, Md 21030
(301) 785-2266

Mindscape, Inc.• A Software Toolworks Company.

Loopz™ is very simple in it's concept and very addictive in it's play. The challenge is to take pieces of different sizes and shapes presented randomly and put them together to form loops. You can move the pieces and rotate them. If you complete a loop, it disappears and you score points. The more complex the loop, the more points. $49.95. **Inquiry # 308**

Days Of Thunder, features the challenge of eight different races on seven different race tracks. Players will be able to choose a car, then customize it for maximum speed and handling on the course. A variety of perspectives, including first-person from behind the wheel, dynamic 3-D driving action and digitized sound give Days Of Thunder an intense realism in a race against the clock. Available by Summer's end. price unavailable. **Inquiry # 309**

Mindscape Inc.
3444 Dundee Road
Northbrook, IL 60062
(708) 480-7667

Spectrum HoloByte

Alexey Pajitnov, the Russian author of Tetris, comes back with a follow-up to his highly successful (and addictive) game, called ***Welltris***. The game is similar to Tetris, except that the blocks now drop straight down, making the game 3-D. $34.95. **Inquiry # 310**

Alexey has another game out called ***Faces (Tris III)***. Pieces of a person's face take the place of blocks. Your goal is to make the face properly as pieces drop. Familiar faces like Gorbachev and Margaret Thatcher come piece by piece. Facial parts are interchangable. Due Soon. $39.95. **Inquiry # 311**

Flight of the Intruder, is now out as a movie and a game. You fly either an A-6 Intruder or F-4 Phantom over Vietnam in raids on SAM sights, ammo dumps and more. Available late this year for $59.95. **Inquiry # 312**

Spectrum HoloByte
2061 Challenger Drive
Alameda, CA 94501
(415) 522-0107

(continued on page 95)

The Command Line

by Rich Falconburg

NOT LONG AFTER I PURCHASED MY FIRST Amiga I began experimenting with serial port communication. Having spent several years working with multiuser systems that use simple alphanumeric terminals as the common I/O device, I was naturally very interested in seeing how the Amiga would perform with a similar configuration. I knew it was capable. I spent some time learning the intricacies of the CLI and discovered that it was possible, in theory at least, to use redirection to create a CLI connection via the SER: device. As many of you may know, this was an exercise in futility, a process I've become something of an expert on. You see, the serial port very generously provides you with a buffer whose size may be changed via the Preferences utility. This helps compensate for the slow output devices that are generally connected to a serial port. For output, this is beneficial in that it frees up the command line, if you did not run the program in the background, and returns the system prompt to you more quickly. This buffer also works the other way. If you send data to the computer via the serial port it will store the information in the buffer until the number of bytes reaches the amount specified by Preferences. The buffer is then sent on to the process expecting the input, emptying the buffer. For proper operation with a terminal, the buffer size would need to be zero, or no buffer at all. However, zero bytes is NOT an option for the serial port. I was able to make it work—marginally. Pressing the RETURN key 500 times is not exactly productive. Undaunted, I continued to peruse the available literature, determined to find a way around this limitation even if it meant that I might have to write a special program to achieve my goal. Much to my chagrin I discovered this "feature" was built into the serial device, meaning that even programming my way around it would be no cake walk. The *ROM Kernel Manual* states that the minimum buffer length for the serial port connection is 512 bytes, period. *How rude*! Having only just begun working with the C programming language, I was not yet ready to tackle the chore of writing a device handler. As a result, my project was tabled for several months and I went on to other things.

Then several months later while scanning the latest list of new programs on my local BBS I came across something that caught my eye. An AUX: handler for using character-oriented devices through the serial port. *Eureka*! I immediately downloaded it and dusted off my terminal and began testing it. It may come as a surprise to some of you, but the AUX: device handler as provided by Commodore is relatively new. They didn't invent the idea, or at least they were not the first to provide it. Steve Drew distributed his AUX: handler long before Workbench 1.3 hit the streets. He has recently upgraded it and included some very nice additions to the package.

→

For those of you who decided to abandon Steve's version in favor of the one shipped by Commodore, I would say, "Take another look." Among other things, his handler has always been more cooperative in allowing you to close the AUX:CLI from another CLI.

This version now includes an AmigaShell-compatible startup script and several support programs. These include:

reqoff: A utility for preventing requestors from popping up from things such as specifying an unmounted volume, effectively locking you out of the system. No problem if you are using it locally but a real pain if you are using a modem. This utility is unique compared to other similar programs in that it only cancels requestors caused by the process from which it is executed.

password: A password-locking facility that will prevent unauthorized access to your Amiga. It allows you to specify the valid password required as a parameter. It does not prompt or echo to the terminal.

emacs: This is version 3.9 of the very capable MicroEMACS editor. Steve has made the necessary changes required to make it work through the AUX:port with any ANSI standard (VT100-compatible) terminal. Only the executable and an .emacsrc file are supplied.

az: This program takes advantage of the XPRZmodem.library to provide a means of downloading while using the AUX: port. The library file is *not* included.

Version 1.1 of Steve's AUX handler provides the following additional features:

Control R key support: This will re-display the current line as it is entered into the computer. Handy if you have a terminal that insists on being dyslexic about the Backspace and Delete keys or other keys that may inadvertently cause the command line to loose its place.

Improved type-ahead: Lines entered while information is being displayed to the screen will not interrupt the output as is the norm for a CLI. VMS users will find both this and the CTRL-R key very comfortable. Also, a tone will sound to alert you that the type-ahead buffer is full. Xon and Xoff (Control S and Control Q) are supported, making remote use very natural and eliminating some of the hazards of the usual method for pausing display output as provided by the console interface.

The handler will also prevent you from invoking multiple AUX: sessions and will allow you to kill off the AUX CLI with a simple

```
1> ECHO >AUX:ENDCLI
```

If anything other than the password program described above is running on the AUX: port, the CLI will not be killed. You may send a line of text to the AUX: session including the output of other commands (handy for sending system messages to a user), but it seems that the only command interpreted by the shell via the AUX handler is the one shown above.

Installing this handler is relatively painless. A new Mountlist entry is provided and instructions for using the new handler are easy to follow. A command script is included which creates several aliases and protects the shell with the password program. It is a simple matter to create your own procedure to call a different shell. I'll give some examples later. I've been using this handler for some time now and find it to be very reliable. It works well with programs that use the standard CON: device for input and output. I especially appreciate the inclusion of the modified EMACS editor which now makes the AUX: port extremely useful. A word of caution: not all shells are created equally and some may not work properly through the AUX: port. Occasionally I've had lockups and GURUs occur which seem to have been related to the shell/AUX combination.

One problem I encountered with the Commodore AUX: handler is its handling of line terminators. I guess Commodore didn't think that anyone would want to use character-oriented programs that assume standard ANSI control codes, such as Carriage Return/Line Feed, which would be used to terminate a line. (Steve's handler makes no such assumption.) As a result, the output of some commands and programs will not format properly with the Commodore AUX: handler. Most notable of these is the SKsh shell discussed last issue. Line Feeds are received but, without the carriage return, every line crowds the last character position on the display. On the other hand, SKsh is a natural combination with Steve Drew's AUX: handler. The two programs complement each other very nicely. Let's examine this a little closer.

First I should mention that Version 1.4 of the SKsh shell is now available and includes some important changes. Here's a quick summary:

Added:

case/esac statement: A multiple test construct.
complist command: An enhancement to the file name completion mechanism.
Resident support: Uses the ARP resident standard.
A TinySKsh: A shell without editing, a history buffer, and several of the built-in commands.
SIZE: A variable for differentiating between a Tiny or normal shell.
LLMIN: A variable that sets minimum length of lines to be added to the history buffer.
MAXDIST: Used to set the maximum distance to search down in the history buffer for a line identical to the one being entered. If one is found, the duplicate line will not be added to the history buffer.
ROOT: A *very* helpful variable that allows you to specify the Root volume which will then be referenced by the single slash ("/") (Unix style root).
LASTRC: A variable that will contain the return code from the last external command.

New external commands:

grep and fgrep: Text-searching commands with options.
view: An extended "file" command.
tee: A very useful piping command that allows for tapping the data stream between two piped commands.
du: Command that will display your disk usage—that is, how many bytes are being used and where.
crc: Command that will calculate codes based on the file data which can be checked after transfer. Handy for noisy communication links.
srun: Another "run in the background" command with a twist: it accepts an input file, an output file, stack and priority arguments, and parameters to be passed to the program being run.

CHANGES

The cp command is now an external command and has some added capabilities.

(continued on page 78)

C NOTES From the C Group

by Stephen Kemp

THOSE OF YOU WHO MISSED LAST MONTH'S COLUMN may not be up to speed on the current subject that I am discussing—doubly linked lists and queuing. To refresh, a linked list is a method of storing information (data) in such a way that one data item "points" to the item that logically follows it. A doubly linked list follows the same principle, but has the added capability of "pointing" to items that occur prior to the current member. This means that you can traverse a list forwards and backwards.

Doubly linked lists revisited.

In the last issue I included code that defined a method of queuing and the associated structures required. In the interest of those who did not see those structures, here they are again:

```
typedef struct x {
   struct x      *prev;
   struct x      *next;
   unsigned short len;
} QITEM;

typedef struct {
   QITEM         *bot;
   QITEM         *top;
   unsigned long  cnt;
} QMAIN;
```

The key things to know about a queue are where the first and last elements of the list are located, and how to reach members that occur before and after any particular element. The easiest way to handle these tasks is through structures like those that I have defined.

The first structure type is named QITEM. Given any member, you have to be able to find the previous member (prev) as well as the member that occurs next. Since the next and previous pointers point to members of the queue, the structure defines these pointers as the same type of structure that we are defining. An additional item is included in this structure to enable the members to handle variable length items. The "len" represents the amount of memory required by the data but does not include the overhead of the QITEM structure. As evident, if each member maintains its QITEM structure it is possible to reach the items that precede and/or follow it by referencing the structure's next or previous pointers.

QMAIN is a header structure for the queue of items. This structure contains a pointer to the first (top) QITEM and a pointer to the last (bot) QITEM. Additionally, the queue header contains a counter (cnt) of the number of items contained in the list. Notice that the order of the structure's "top" and "bot" is important. By aligning "bot" in the same position as "prev" in the QITEM structure and "top" with the "next" position, it is possible to make a complete "circle" through the items.

With the structures that we have defined, it is now possible to write code to handle doubly linked lists. Last month we covered the basic queuing functions that initialize and free a queue, add items to and remove items from a queue, and some important positioning functions. This month, I want to introduce a new function.

The function is named "qcut". This function will "move" everything in a queue—beginning at a designated element—into another queue. The code for the function follows:

```
/*-------------------------------------*/
/* QCUT is used to move the elements from one queue to another
*/
/* beginning at the QITEM pointer that was passed.            */
/* The new queue is assumed to be empty.                      */
/*-------------------------------------*/
void qcut(QMAIN *src,QITEM *member,QMAIN *dest)
{
    QITEM *ptr;
    unsigned long cnt;

    cnt = 0;                          /* first count the members
*/
     for(ptr = member; ptr != (QITEM *)src; ptr = ptr->next,
cnt++);

   if (cnt == 0){                     /* if no members included
*/
        qinit(dest);                     /* just init the dest
properly*/
    }else{
       dest->top = member;            /* first element is this
one */
       dest->bot = src->bot;           /* last element is this
one  */
       dest->cnt = cnt;                /* this is the count */

      src->bot = member->prev;        /* adjust bottom of orig
queue*/
       src->bot->next = (QITEM *)src;   /*  now point to main
*/
      src->cnt -= cnt;                /* adjust orig queue count
*/

       member->prev = (QITEM *)dest;    /* this guy now point
to dest*/
      dest->bot->next = (QITEM *)dest; /* the last one too */
    }
}
```

As I have mentioned before, one of the advantages of these types of queues is that you don't have to actually move data around once it is stored. The actual "sequence" of data is maintained by the pointers in the queue, and where the data actually resides is secondary. To those who missed last month's column, the elements are stored in memory with a QITEM structure first, and the data follows. As already stated, once the data is stored, the pointers maintained in the QITEM structure determine the actual queue sequence.

The first thing the function does is determine how many elements are going to be involved. Since the QMAIN structure maintains a count of elements found in the queue, it is necessary to know how many are being removed from the source queue and, likewise, how many end up in the destination queue. All you have to do to count the pointers is "next" through the pointers until you are pointing at the source queue's header. Since you are comparing a QITEM pointer, you have to cast the queue's structure as though it were one too.

If you determine that no elements are going to be moved, all that is required is to initialize the destination queue. This is important since the function that calls qcut will probably expect that the destination queue is now properly established. Remember, initializing a queue simply involves pointing the top and bot pointers in the QMAIN structure to the QMAIN (it points to itself). Naturally, the count is also set to zero.

Should the function decide that elements will be "moved", then a different path is followed. First, the QMAIN structure for the destination queue has to be established. The top pointer will be set to the member that was passed as a parameter. The bot pointer can be taken from the source queue since it has already been established there. And the count is assigned the number of elements that were involved.

Now it is time to adjust the source queue so that its integrity is maintained. The last element of the source queue will be the element that occurred prior to the member passed. After establishing the new bottom of the source queue, you can use that pointer to make the last element (which still points to the member being removed) point to the header of the source queue. Once the count of the source queue has been adjusted, the integrity of the source queue has been reestablished.

In a similar manner, you have to complete the integrity of the destination queue. Although the QMAIN structure has been completed, you still have to adjust the member passed so that its previous pointer references the QMAIN of the destination queue. Likewise, you also have to make sure the last element's next (which still points to the source queue's QMAIN) now references the destination queue's QMAIN.

After all these things have been completed, and executed, you will have two independent queues that were once one. No data has actually been moved, which means that this type of operation will almost be instantaneous.

Get the code from last month's column and put this function into operation. If you don't have the issue, order it elsewhere in this issue and give queuing a try. It probably won't take you long to think up a number of uses for queuing.

•AC•

---

*(**Command Line**, continued from page 76)*

The backslash is now used to override the special characters, allowing them to be used in text strings. This is more compatible with the Unix method than the double quoting required by SKSH 1.3.

The filename completion mechanism now handles wild card characters properly.

The variable substitution mechanism has been improved and several other bugs have been fixed including a frustrating one in the date command. Also, the code size has been reduced, as have the stack requirements.

CUSTOMIZING AUX

The easiest way to use the SKsh shell in place of the AmigaShell is to create a file that includes the following:

```
STACK 10000
SKSH
```

This is the minimum you'll need. Add what other embellishments you wish to execute. Just be sure to place them before the SKSH command as this command takes over the CLI. Now substitute the following for the NewShell command in the aux-startup script included with the Drew AUX: handler

```
NEWCLI From SKsh_AUX-startup
```

where SKsh_AUX-startup is the name of the file created above. I would recommend creating SKsh scripts and using them rather than executing commands from either of the above files for setting up your environment. The reason should be obvious. SKsh provides a far more robust command set for scripts than does the AmigaShell. One very nice feature about this shell that I neglected to mention is that the command line editing is a subset of the EMACS control keys. Even if you have a terminal that does not support ANSI sequences you will still be able to use the line-editing features with CONTROL key combinations. Using it with a number of the supported keys will provide you with complete line editing capabilities, even on a "dumb" terminal.

There is a wonderful world to explore in remote communications with the Amiga. With the advent of multi-port serial boards it is now possible to create a full-fledged multiuser system. There are a few important additions that need to be made to the Amiga's command environment to fully support this kind of operation, such as special use of permission bits to control access, resource tracking facilities, user mail, and more. Most of the software you need can be found on your nearest Bulletin Board System.

Next time I will begin a look at ways to connect Amigas together to form a powerful operating environment that allows sharing resources between machines in ways that would make owners of those other computers turn green.

•AC•

Send questions or comments to Rich Falconburg, c/o Amazing Computing, P.O. Box 869, Fall River, MA 02722-0869, or send Email to R.Falconburg on GEnie.

*(**Trees**, continued from page 60)*

The code inside the inner 'if' statement creates a first-in, first-out list (alias a queue). The queue node is allocated and its node list pointer is set to point to the child list of the node being visited. Its 'next' pointer is set to NULL, allowing the end of the queue to be detected. If the new queue node is not the first one allocated on the level being visited, it is linked in as the 'next' queue node in the linked list. Otherwise, it is assigned as the top QNode pointer and becomes the first node in the queue. The new queue node pointer is then saved as the previous queue pointer, so the queue nodes can be linked together. What is actually being built is a list of lists. The queue nodes are the list header nodes, and the linked lists of sibling nodes are the pointed-to lists. The queue nodes are linked together to build the entire search queue for the next level.

The nodes are actually visited in the innermost 'while' loop by simply following the 'next' pointers in the list of sibling nodes. When the end of a sibling list is reached, the innermost 'while' loop ends. The outermost 'while' loop iterates on the linked list of queue nodes which was sent in to the procedure. The list of nodes attached to each queue node is searched. After the list is searched, the next queue node in the list is accessed, and the queue node that was just used is freed, since it is no longer needed.

This nested iteration, queue node allocation and freeing continues until the end of the queue is reached. At that time, if a new queue has been constructed, the procedure is recursively called to traverse the nodes for the next level. Otherwise, the procedure returns from the recursion. The traversal is started by forming a queue consisting of only the root node and calling the traversal procedure with the queue pointer as the input argument.

As you can see, breadth-first traversal requires a lot of internal gyrations and housekeeping. So why use it? Actually, I hardly ever use it, but I'll return to this question after describing how the search is done.

TREE SEARCH

Trees are used to store data. Found data has to be stored sooner or later. Trees are also used to model many kinds of problems. 'Search' is a general term which describes different methods of finding a particular piece of data, finding a path from one place to another, or finding a solution to a problem represented as a data structure. While there are many different techniques for searching a tree, most of the techniques are modified or enhanced versions of depth-first traversal or breadth-first traversal. These techniques can be grouped into three categories. The first category is searching just to find any satisfactory path. Depth-first search and breadth-first search belong in this category. A second type is searching to find a 'best' path which meets some condition of optimality, such as finding the shortest path from one node to another. A third type of search is game tree. A game tree search is a specialized type of tree used to (what else !!) represent games.

Both depth-first search and breadth-first search are modifications of the traversal techniques discussed above. If you know how to traverse a tree, you already know how to search the tree. The difference between traversal and search is that in search, instead of just visiting a node, the node is examined to see if it or its data matches some search criteria. If a match is found, the search ends and a pointer to the found node is returned. If the entire tree is traversed and no node is found, a NULL pointer is returned to indicate that the search had failed.

Depth-first search is one of the simplest search techniques for trees. In pseudo-code, a recursive algorithm for depth-first search of a natural binary tree implemented as a list of lists is:


```
procedure dfSearch( node )
   while node is not NULL
      if node is GOAL, return node
      else
      if node->child is not NULL
         return dfSearch( node->child )
      else
         set node to node->next
   end while
end dfSearch
```

This algorithm is implemented in C as the procedure 'dfSearch()' in the demonstration program.

Just like breadth-first traversal, breadth-first search is simple to understand, but it is difficult to implement because the nodes on a level may not all be in the same sibling list. The pseudo-code algorithm for a recursive breadth-first natural binary tree search is: `procedure bfSearch( queue )`

```
   while queue is not empty
      get the first node list from the queue
      while node list is not empty
         if node is GOAL, return node
         else
         if node->child is not NULL,
```

List Of Advertisers

Need more information?
Need help?
Need to know?

Contact the AC advertisers!

Please use the Reader Service Card to contact those advertisers who have sparked your interest. Advertisers want to hear from you. This is the best way they have of determining the Amiga community's interests and needs. Take a moment and contact the companies with products you want to know more about. And, if you wish to contact an advertiser directly, please tell them you saw their advertisement in

Amazing Computing
For The Commodore Amiga

THE NEW AmiEXPO

THE AMIGA PERSONAL COMPUTER SHOW

ALL NEW SEMINARS!

SEE THE LATEST AMIGAS

October 5-7 at The Disneyland Hotel
Anaheim, California

Sponsored by AMIGA WORLD

Admission includes the Exhibition, Seminars, Keynotes & Amiga Artists Theatre!

120 Amiga Exhibitors Featuring State of the Art Software and Hardware, at the lowest prices!

Master Classes Available in Amiga Graphics, Video, MultiMedia, Animation, Rendering and Publishing!
Seating for Master Classes is limited; call for schedule and availability before registering.
PRE-REGISTRATION DEADLINE IS SEPTEMBER 21, 1990
(No cancellations or refunds after deadline)

For Hotel Reservations Call the Disneyland Hotel at (714) 778-6600
Hotel reservations deadline: September 19, 1990
For discounted airfares, call American Airlines at (800) 433-1790 and give them this ID: 12Z 04F

REGISTER TODAY!

Register by Mail using the coupon below or Call 800-32-AMIGA Nationwide (or 914-741-6500)
For Your Ticket to The Amiga Event!

Yes, I want to come to AmiEXPO-California

___ Friday ___ Saturday ___ Sunday

Registration is $5 Additional At The Door

One day - $15 _____
Two days - $20 _____
Three days - $25 _____

Master Class(es) - List Class and Time - $60 Each

_____ _____
_____ _____
_____ _____
_____ _____

NFAW

Total Amount Enclosed _____

NAME _____
COMPANY _____
ADDRESS _____
_____
CITY _____ STATE _____ ZIP _____

For ___ MasterCard or ___VISA Payment
Expiration Date _____
Account Number _____
Name as it appears on card: _____
Signature _____

Make Check or Money Order Payable to:
AmiEXPO 465 Columbus Ave., Ste. 285
Valhalla, NY 10595

Circle 119 on Reader Service card.

```
            add node's child list to the next-level queue
        set node to node->next
      end while
    set queue to queue->next
    end while
    return bfSearch( next-level queue )
end bfSearch
```

In the demonstration program, the procedures bfSearch(), allocQueue(), and freeQueue() implement the breadth-first search algorithm.

Breadth-first search requires a lot more housekeeping than depth-first search, and on the surface, it would seem to be less efficient than depth-first search. This is not necessarily true! The really important measure of the efficiency of a search is how fast can a node be found. This efficiency depends more on how many nodes have to be searched than anything else. The shape of the tree being searched and the location of the node in the tree has a much greater effect than anything else on how efficient one search technique is over the other. If the tree being searched does not have very many levels, but it has many nodes on each level, a depth-first search may be more efficient than a breadth-first search. Likewise, if the tree has many levels but not many modes on each level-breadth-first search may be efficient. For the tree in the demo program, the location of the node being sought has the largest effect on the search. What this boils down to is that no single search technique is best for all problems. A technique which works great for one set of data may not be as good for another set of data!

The demo program will allow you to watch how both types of search move from node to node, generating a trace of the searches. It's interesting to use both techniques to search for the same node and compare the number of nodes each technique examined before finding the node being sought. You will be able to clearly see the effect on the search of the node's position in the tree. Listing 1 is an example of the trace output from the demo program. As you can see from the listing, depth-first search examined six nodes before finding the goal node, and breadth-first search only examined four nodes before finding the same goal node.

Neither of these search techniques has any 'smarts'. Both just start at any node and search every node which can be reached from that start node. As mentioned above, many modifications have been made to both breadth-first and depth-first search to make them better. Most of these modifications involve either modifying the search order by trying to decide which node is the 'best' node to search next, or by 'pruning' the tree to eliminate nodes from even being searched. In applications where the data can be ordered in some way (such as alphabetically), the tree is built from the start to allow efficient search. If you are interested in learning more about other types of search or search optimization, the references at the end of the article will get you started in your 'search' for information.

THE SEARCH DEMONSTRATION PROGRAM

The demonstration program in listing 2 shows how depth-first search and breadth-first search are implemented in C, and it demonstrates how the searches work. The program is interactive and lets you pick a search technique and a node to search for. You then step through the search by clicking the left mouse button. The program also uses some 'advanced' techniques. Even function pointers and pointers to pointers are used! The program is compiled and linked with the Lattice command 'lc -L searchTree.c', and is run from the CLI.

This is a long program to type in, but if you are interested in using recursion or recursive data structures in your programs, it includes most of the basic C techniques you need to know how to get started. If you are just beginning to use the Amiga's graphics and Intuition procedures, the program will give you a good idea of how to call some of the graphics, text, menu routines, and how to process Intuition input events. Many of the procedures can be easily modified and used in other programs, especially the Intuition-related procedures.

The program constructs the binary tree shown in figure 2 and displays the tree in a window in the workbench screen. A small menu consisting of 'PROJECT - QUIT', 'PERFORM - DFSearch' and 'PERFORM - BFSearch' is attached to the window. The program can be stopped by picking the window close gadget or by picking the 'PROJECT - QUIT' menu command. A message is displayed in the lower left corner of the window which tells how to search for a node, or tells which node was found after the search was completed. Either breadth-first search or depth-first search can be performed by first selecting the type of search from the 'PERFORM' menu, then clicking the select button over a search node. Both types of searches look for a node at the coordinates where the mouse cursor was located when the select button was pressed and released. When the search begins, the search coordinates are displayed in the message area. If a node is found, the level number and sequence number (for example, 3.7) of the found node is displayed. When the search ends, the program will let you search for another node using the same type of search, or you can pick another search type and another node to search for.

The program writes a trace of the recursive calls which each search procedure makes as it traverses the tree. The trace output is written to a file named 'searchTree dta' in the current directory. If you are having trouble understanding how the searches work, or are confused by recursion, comparing the trace output with the tree diagram should help.

The main part of the demo program was described in my last article, but some new procedures have been added. All of the procedures contain a lot of comments, so you should be able to figure out how the procedures work. The following paragraphs describe the program logic.

The procedure main() implements the top-level program flow and calls AskFont() to get information about the window's font height. It builds the tree by initializing a static root node, then calling allocNode() and placeNode() to recursively generate all of the tree nodes, locating them at the correct positions. A window is allocated, initMenu() is called to create the menu, and the tree is recursively displayed with a call to displayTree(). Finally, procedure handleInput() is called to process input events. When handleInput() returns control to main(), cleanUp() is called to close the window, free all structures, and close all files. The font height is used to size the menu items and to position the message text.

The procedures which take care of the menus are initMenu(), allocMenu(), allocMenuItem(), allocIText(), and freeMenu(). Intuition's menus are lists of lists, and initMenu() dynamically allocates the structures, initializes them, and links them together. The top-level menus are allocated and initialized in allocMenu() and the menu items are allocated and initialized in allocMenuItem(). Since

menu items use Intuitext structures, allocIText() is called to allocate and initialize the menu item text structures. When the program ends, freeMenu() is called from cleanUp() to recursively free all of the menu structures using depth-first traversal. Incidentally, freeMenu() can be modified to free any dynamically-allocated menu strip by adding an inner 'while loop' to take care of freeing any menu subitem lists.

The depth-first search is done by two procedures: doDFSearch() and dfSearch(). DoDFSearch() is called (using a function pointer) from handleInput() and is sent the coordinates of the location where the mouse 'select' button was released. DoDFSearch() modifies Intuition's communications port to stop all event messages from arriving except for mouse button messages. It then displays an information message in the search window and calls dfSearch() to perform the actual search. After dfSearch() returns, a message with the search result is written in the tree window, and waitForButton() is called. WaitForButton() just waits until a mouse button is pressed and released, then returns. DoDFSearch() then restores the IDCMP and returns to handleInput().

Breadth-first search is done by doBFSearch() and bfSearch(). DoBFSearch() is almost identical to doDFSearch(), except that a queue made of one queue node is allocated and sent to bfSearch(). The queue node points to the single root node of the tree.

The search procedures, dfSearch() and bfSearch() are similar to the traversal procedures described above. The main difference is that trace code, graphics display code, and code to see if the search coordinates are inside each node has been added. The procedures are recursive, and recursion ends either when a node is found or when the entire tree has been searched. To make bfSearch() less confusing, the code to allocate and link a new queue node into the search queue is in allocQueue(), and the code to free the queue space is in freeQueue(). Since allocQueue() needs to modify the pointers sent to it, the addresses of the pointer variables 'topQueue' and 'prevQueue' are passed to allocQueue(). In allocQueue(), these variables are declared as pointers to pointers (prefixed with '**'). When I first started using pointers, it took me a long time to figure out how to declare pointers in a procedure, so that the changes to the pointer could be sent back to the calling procedure. The call to allocQueue() and the pointer declarations in allocQueue() show how it's done.

AllocQueue() just sets up a new queue node and adds it to the end of the queue. FreeQueue() frees the entire linked list of queue nodes by iterating using 'next' pointers.

The entire tree is built, and the nodes are placed with allocNode() and placeNode(). The tree is recursively freed with freeNode(). The tree is graphically displayed using displayTree() and displayNode(). In my last article, I described how these procedures worked . If you missed the article, you should be able to figure them out from the comments in the code.

All Intuition input is taken care of in handleInput() and waitForButton(). HandleInput() calls one of the two search procedures using a function pointer. The function pointer value is set to point to either doDFSearch() or doBFSearch(), depending on which menu item was picked. After the mouse select button is released, handleInput() checks to see if the function pointer is valid, and if it is, calls the search function through the pointer. When either the 'QUIT' menu item is picked or the window 'CLOSE' gadget is picked, handleInput() makes sure all messages are given a reply, then returns control to main(). The program then cleans up behind itself and ends.

Most of the program is fairly straight-forward, although recursion and recursive data structures can be confusing in even very simple programs. I hope the comments in the code and the algorithm descriptions above are clear enough for you to understand how the program works, because the techniques are useful for Amiga programs. The program shows how Intuition's menus can be managed with tree techniques. If you look at window and gadget structure definitions, you'll see that these structures are also lists which are combined into lists of lists. In fact, almost all Amiga data structures are lists and lists of lists! These structures are so important to the Amiga that the 'exec' library has list management procedures built right into it.

Pointers, recursion, linked lists, lists of lists, and trees are techniques and data structures which are confusing when you first start working with them, but once you understand the basic ideas, they become easy to use. From my own experience, I find the best way to learn new programming methods is by copying and modifying plenty of examples of the working code. With a little persistence (and a lot of debugging!) these structures and techniques can be mastered and used to create some Amazing Amiga programs.

FURTHER READING

If you are interested in different types of search, *Artificial Intelligence,* by Patrick H. Winston, is a good book to start with. The book is published by Addison-Wesley Publishing Company, Reading, Massachusetts. Many types of 'search' are described in an English-like pseudo-code, and the descriptions uses a lot of clear

diagrams. This book is a good and understandable introduction to most of the ideas being explored in AI research.

Artificial Intelligence Using C, by Herbert Schildt, has a chapter dedicated to several common search techniques, and has examples of C programs which use the techniques. The book is published by Osborne McGraw-Hill, Berkeley, California. This is also a good introduction to AI, and each topic covered has working C programs which demonstrate the concepts presented.

For those of you who are interested in graph theory (trees are special types of graphs) and like to read mathematical texts, *Concepts in Discrete Mathematics*, by Sartaj Sahni, has a chapter which covers graphs, trees, paths, and connectivity. This book also covers many of the mathematical ideas used in theoretical computer science. The book is published by the Camelot Publishing Company, Fridley, Minnesota.

LISTING ONE

```
/*=========================================================*/
 *  Listing 1.  searchTree.dta - example trace output from
 *              searchTree program.  Both depth-first and
 *              breadth-first search were performed by
 *              selecting node 2.1.
/*=========================================================*/

Performing depth-first search...
Seeking node at 187,101
In DFSearch at level 0
Searching node 0.0
   In DFSearch at level 1
   Searching node 1.0
      In DFSearch at level 2
      Searching node 2.0
         In DFSearch at level 3
         Searching node 3.0
         Searching node 3.1
         Searching node 3.2
         Returning from level 3
      Searching node 2.1
   Returning from level 1
Returning from level 0

Depth-First search complete...
found node 2.1

Performing breadth-first search...
Seeking node at 186,100
In BFSearch at level 0
Searching node 0.0
   In BFSearch at level 1
   Searching node 1.0
   Searching node 1.1
      In BFSearch at level 2
      Searching node 2.0
      Searching node 2.1
      Returning from level 2
   Returning from level 1
Returning from level 0

Breadth-First search complete...
found node is node 2.1

Done..
```

LISTING TWO

```
/*=========================================================*/
/*
 *  Listing 2 - searchTree.c
 *              program to demonstrate tree traversal
 *              and tree search
 *
 *  Lattice compile & link command:  lc -L searchTree.c
 *
 *  copyright 1989 by Forest W. Arnold
*/
/*=========================================================*/

#include <exec/types.h>
#include <intuition/intuition.h>
#include <stdio.h>

/*  intuition stuff                                        */

#define INTUITION_REV   34
#define GRAPHICS_REV    34
struct IntuitionBase *IntuitionBase;
struct GfxBase       *GfxBase;

/*  my 'make-life-easy' macros                             */

#ifndef NOT
#define NOT     !
#endif
#define NEW(x)  ( (x *)malloc( sizeof(x) ) )

/*
 *  Define the nodes and queue structures & types
*/

typedef struct node        /*  the tree nodes             */
{
   struct node    *next;   /*  sibling link               */
   struct node    *child;  /*  child link                 */
   int            x1,y1;   /*  left, top coordinate       */
   int            x2,y2;   /*  right, bottom coordinate   */
   int            level;   /*  level in the tree          */
   int            seqNo;   /*  sequence within the level  */
}  NODE_T;

typedef struct queNode          /* used for queue for      */
{                               /* breadth-first search    */
   struct queNode   *next;      /* sibling link            */
   NODE_T           *nodeList;  /* list header             */
}  QNODE_T;

/*  define the output file                                 */

char *outFile = "searchTree.dta";
FILE  *otp = NULL;

/*
 *  define node geometric dimensions & horiz and vert
 *  spacing between nodes
*/

#define HRAD      15        /*  ellipse horizontal radius */
#define VRAD      8         /*  ellipse vertical radius   */
#define HDIST     20        /*  horiz dist between nodes  */
#define VDIST     20        /*  vert dist between nodes   */
#define HCONS     (HRAD+HRAD+HDIST) /* horiz & vert dist */
#define VCONS     (VRAD+VRAD+VDIST) /* actual values     */

/*  global variables                                       */

static struct Window   *window = NULL;
static struct RastPort *rp = NULL;
static int             textHeight = 0;

/*  intuition interface procedures                         */

struct Window     *displayWindow( int l,int t,int w, int h,
                                  char *name );
struct IntuiText  *allocIText( short l,short t,
                               unsigned char *strPtr );
struct MenuItem   *allocMenuItem( short l,short t,
                                  short w, short h,
                                  struct IntuiText *strPtr );
struct Menu       *allocMenu( short l,short t,
                              short w,short h,
                              unsigned cnar *strPtr );
struct Menu       *initMenu( void );
void              freeMenu( struct Menu *topMenu );

/*  depth-first search procedures                          */

void    doDFSearch( NODE_T *root,int x,int y );
NODE_T  *dfSearch( NODE_T *topNode,int indent,
                   int x,int y );

/*  breadth-first search procedures                        */

void    doBFSearch( NODE_T *root,int x,int y );
NODE_T  *bfSearch( QNODE_T *queue,int indent,int x,int y );
QNODE_T *allocQNode( QNODE_T **topQNode,
                     QNODE_T **prevQNode, NODE_T *node );
void    freeQueue( QNODE_T *topQNode );

/*  tree procedures                                        */

NODE_T  *allocNode( int number,int level );
int     placeNode( NODE_T *topNode,int top,int left );
void    freeNode( NODE_T *topNode );
void    displayTree( NODE_T *topNode );
void    displayNode( NODE_T *node,int pn,int mode );

/*  input procedures                                       */

void    handleInput( NODE_T *rootNode );
void    waitForButton( void );

/*  utility procedures                                     */

void    displayMsg( int x,int y,char *wMsg1,char *wMsg2,
                    char *wMsg3,int pen );
void    cleanUp( NODE_T *root,struct Menu *menu );
void    closeLibs( void );
void    trace( char *msg,int indent );
int     openLibs( void );

/*=========================================================*/
```

```
void main( int argc,char** argv )
{
   NODE_T           root;          /* top-level node    */
   struct TextAttr  taData;        /* window's text data*/
   struct Menu      *menu;         /* search top menu   */

   /* open files and libraries                          */

   if ( outFile[0] != '\0' )
   {
      if ( NOT (otp = fopen(outFile,"w")) )
      {
         printf("\nCan't open trace output file...\n");
         exit(1);
      }
   }
   if ( NOT openLibs() )
   {
      cleanUp( NULL,NULL);
      printf("\nCan't open system libraries...\n");
      exit(1);
   }

   /* set up the top-level node of the tree             */

   root.level = 0;
   root.seqNo = 0;
   root.x1 = 20;  root.y1 = 15;
   root.x2 = root.x1 + HRAD + HRAD;
   root.y2 = root.y1 + VRAD + VRAD;
   root.next = NULL;

   /* recursively allocate tree & position nodes        */

   root.child = allocNode(2,1);
   (void)placeNode(root.child,root.y2+VDIST,root.x1);

   /* allocate window & menus                           */

   if ( NOT (window = displayWindow(0,0,640,200,
                                    "TREE SEARCH")) )
   {
      printf("Can't allocate display window\n");
      cleanUp( &root,NULL );
      exit(1);
   }
   rp = window->RPort;

   /* get the font height                               */

   AskFont(rp,&taData);
   textHeight = (int)taData.ta_YSize;

   if ( NOT (menu = initMenu()) )
   {
      printf("Can't allocate menus\n");
      cleanUp( &root,NULL );
      exit(1);
   }
   SetMenuStrip( window,menu );    /* attach menu strip  */

   /* recursively display the tree                      */

   displayTree( &root );

   /*  handle menu input until "QUIT" picked            */

   handleInput( &root );
   cleanUp( &root,menu );
   exit(0);
}
/*=====================================================*/
/*
 *  doDFSearch() - driver routine to perform depth-first
 *  search.  Sets up IDCMP & displays message, then calls
 *  search routine.
*/
/*=====================================================*/

void doDFSearch( NODE_T *root,int x,int y )
{
   char              msg[80],*wMsg1,*wMsg2,*wMsg3;
   NODE_T            *found;

   /*
    *  set IDCMP to get mouse button events, then print
    *  messages & call DF search procedure
   */

   ModifyIDCMP( window,MOUSEBUTTONS );

   sprintf(msg,"\nPerforming depth-first search...");
   trace( msg,0 );
   sprintf(msg,"Seeking node at %d,%d",x,y);
   trace( msg,0 );
   wMsg1 = "Depth-First Search:";
   wMsg2 = msg;
   wMsg3 = "'select' to search next node";
   displayMsg( 20,160,wMsg1,wMsg2,wMsg3,1 );

   found = dfSearch( root,0,x,y );

   displayMsg(20,160,NULL,wMsg2,wMsg3,0);
```



```
   sprintf(msg,"\nDepth-First search complete...");
   trace( msg,0 );

   if ( found )
      sprintf(msg,"found node %d.%d",
                  found->level,found->seqNo);
   else
      sprintf(msg,"no node found at search coordinates.");

   trace(msg,0);
   wMsg2 = msg;
   wMsg3 = "'select' to continue";
   displayMsg(20,160,NULL,wMsg2,wMsg3,1);
   waitForButton( );

   /* clear message, restore IDCMP, & return            */

   displayMsg(20,160,wMsg1,wMsg2,wMsg3,0);
   ModifyIDCMP( window,CLOSEWINDOW|MENUPICK|MOUSEBUTTONS );
}
/*=====================================================*/
/*
 *  dfSearch() - recursively perform depth-first search.
 *  This procedure highlights the node, waits for select
 *  button, unhighlights the node, and recursively
 *     top = topNode;
   sprintf(msg,"In DFSearch at level %d",top->level);
   trace(msg,indent);

   while( top )
   {
      sprintf(msg,"Searching node %d.%d",top->level,
                                         top->seqNo);
      trace(msg,indent);

      /* highlight node, wait for button & unhighlight it*/

      displayNode( top,2,1 );
      waitForButton();
      displayNode( top,2,1 );

      /*  see if node is at the input search coordinates */

      if ( top->x1 <= x && x <= top->x2 &&
           top->y1 <= y && y <= top->y2   )
         return( top );                  /* found it */
```

```
      /* recursively search child nodes                    */

      level = top->level;

      if ( top->child )
      {
         if ( found = dfSearch(top->child,indent+3,x,y) )
         {
            sprintf(msg,"Returning from level %d",level );
            trace(msg,indent);
            return( found );
         }
      }
      top = top->next;     /* get sibling node             */
   }
   sprintf(msg,"Returning from level %d",level);
   trace(msg,indent);
   return( NULL );         /* didn't find a node           */
}
/*=====================================================*/
/*
 *  doBFSearch() - driver routine to perform breadth-first
 *  search.  Sets up IDCMP & displays message, then sets
 *  up a queue structure (first-in, first-out list).
 *  The queue is used for the search, since it links
 *  together disjoint sibling lists on same tree level.
*/
/*=====================================================*/

void doBFSearch( NODE_T *root,int x,int y )
{
   QNODE_T           *rootQNode; /* links lists together */
   NODE_T            *found;     /* ptr to the found node*/
   char              msg[80],*wMsg1,*wMsg2,*wMsg3;

   /* set IDCMP to get mouse button events & show message*/

   ModifyIDCMP( window,MOUSEBUTTONS );

   /* set up the search queue (FIFO list) which chains   */
   /* sibling lists at same level together               */

   if ( NOT (rootQNode = NEW(QNODE_T)) )
   {
      printf("Can't allocate memory for QNode.\n");
      return;
   }
   rootQNode->next = NULL;
   rootQNode->nodeList = root;

   /* output message data & do the search                */

   sprintf(msg,"\nPerforming breadth-first search...");
   trace( msg,0 );
   sprintf(msg,"Seeking node at %d,%d",x,y);
   trace( msg,0 );
   wMsg1 = "Breadth-First Search:";
   wMsg2 = msg;
   wMsg3 = "'select' to search next node";
   displayMsg( 20,160,wMsg1,wMsg2,wMsg3,1 );

   found = bfSearch( rootQNode,0,x,y );

   displayMsg(20,160,NULL,wMsg2,wMsg3,0);
   sprintf(msg,"\nBreadth-First search complete...");
   trace( msg,0 );

   if ( found )
      sprintf(msg,"found node is node %d.%d",
                  found->level,found->seqNo);
   else
      sprintf(msg,"no node found at search coordinates.");

   trace(msg,0);
   wMsg2 = msg;
   wMsg3 = "'select' to continue";
   displayMsg(20,160,NULL,wMsg2,wMsg3,1);
   waitForButton();

   /* clear message, restore IDCMP, & return             */

   displayMsg(20,160,wMsg1,wMsg2,wMsg3,0);
   ModifyIDCMP( window,CLOSEWINDOW|MENUPICK|MOUSEBUTTONS );
}
/*=====================================================*/
/*
 *  bfSearch() - recursively perform breadth-first search
 *  This procedure iterates on a queue containing pointers
 *  to all lists at the same level of the tree.  Each
 *  list is removed from the queue, and the procedure
 *  iterates on each node in the list.  Each node is
 *  displayed, then if the node has a child list, a queue
 *  node is allocated to point to the child list, and the
 *  queue nodes are linked together.  After all
 *  lists at a given level have been searched, the
 *  procedure recursively calls itself to search the next
 *  level using the queue constructed for the next level.
*/
/*=====================================================*/

NODE_T *bfSearch( QNODE_T *queue,int indent,int x,int y )
{
   /*
    *  these qNode pointers are used to construct & link
    *  together the queue of child lists and to iterate
    *  on the current queue of lists
    */

   QNODE_T *topQNode,*newQNode;
   QNODE_T *prevQNode,*nextQNode;
   NODE_T  *found;          /* the found node, if any     */
   NODE_T  *top;            /* the current display node   */
   int     level;           /* current tree level         */
   char    msg[80];

   topQNode = prevQNode = NULL;  /* init next level queue*/
   found = NULL;
   level = queue->nodeList->level;
   sprintf(msg,"In BFSearch at level %d",level);
   trace(msg,indent);

   /*
    *  iterate on the queue nodes for this level.  Get the
    *  list of sibling nodes from the queue, and then
    *  iterate on the nodes in the list.
    */

   while( queue )              /* iterate on queue nodes  */
   {
      top = queue->nodeList;   /* the list of siblings    */

      while( top )             /* iterate on nodes        */
      {
         displayNode( top,2,1 );
         waitForButton();
         displayNode( top,2,1 );

         sprintf(msg,"Searching node %d.%d",
                                 top->level,top->seqNo);
         trace(msg,indent);

         /*  see if node is at the search coordinates    */

         if ( top->x1 <= x && x <= top->x2 &&
              top->y1 <= y && y <= top->y2   )
         {
            /*
             *  Found it.
             *  We need to free any qNode space for both
             *  the current level and also the space for
             *  the next level's queue before returning
             */

            freeQueue( topQNode ); /* child list queue   */
            freeQueue( queue );    /* this level's queue */
            sprintf(msg,"Returning from level %d",
                                       top->level);
            trace(msg,indent);
            return( top );    /* search is over          */
         }

         /*
          * if this node has a child list, allocate a
          * qNode, set its pointer to the list, and link
          * it into the queue for searching the next level
          */

         if ( top->child )
            newQNode = allocQNode(&topQNode,&prevQNode,
                                                top->child);

         if ( NOT newQNode )
         {
            printf("bfSearch:  out of memory\n");
            return( NULL );
         }
         top = top->next;  /* next node in current list  */
      }

      /*
       *  It wasn't in list just looked at - get the next
       *  queue node and free the one just used.
       */

      nextQNode = queue->next;   /* get sibling queue     */
      free( (void*)queue );      /* free last queue node  */
      queue = nextQNode;         /* set current que node  */
   }

   /*
    *  if queue of child lists constructed, call self to
    *  search it
    */

   if ( topQNode )
      found = bfSearch(topQNode,indent+3,x,y );

   sprintf(msg,"Returning from level %d",level);
   trace(msg,indent);
   return( found );  /* either NULL or a found node      */
}
/*=====================================================*/
/*
 *  allocQNode() - allocate & link a new qNode structure
 *                 into a queue.  Pointers to pointers
 *                 are used so the pointer's value can
 *                 be changed
```

```
*/
/*=======================================================*/

QNODE_T *allocQNode( QNODE_T **topQNode,
                     QNODE_T **prevQNode,NODE_T *node )
{
   QNODE_T  *newQNode;

   /*  allocate the new qNode struct                     */

   if ( NOT (newQNode = NEW(QNODE_T)) )
      return( NULL );

   /*  init the links                                    */

   newQNode->nodeList = node;  /* pointer to node list   */
   newQNode->next = NULL;      /* link for queue nodes   */

   /*
    * if a queue has already been started, just link this
    * one in at the end of the queue.  Otherwise, this is
    * first node in the queue
   */

   if ( *prevQNode )
       (*prevQNode)->next = newQNode;
   else
       *topQNode = newQNode;

   *prevQNode = newQNode;   /* ptr to last one in queue  */
   return( newQNode );
}
/*=======================================================*/
/*
 *  freeQueue() - free an entire queue
*/
/*=======================================================*/

void freeQueue( QNODE_T *topQNode )
{
   QNODE_T *prevQNode;

   while( topQNode )  /*  standard linked list iteration */
   {
      prevQNode = topQNode;
      topQNode = topQNode->next;
      free( (void *)prevQNode );
   }
}
/*=======================================================*/
/*
 *  allocNode - recursively allocate & initialize
 *     tree nodes.  After each node at a level is
 *     allocated, this routine calls itself to allocate
 *     the child nodes for the next level.  This is
 *     specialized for making the demonstration tree.
*/
/*=======================================================*/

NODE_T *allocNode(int number,int level)
{
   static int seq[] = {0,0,0,0}; /* count of nodes by lvl*/
   NODE_T     *node,*prevNode,*topNode;
   int        i;

   topNode = prevNode = NULL;

   /* iterate on number of requested nodes for the level */

   for( i = 0; i < number; i++ )
   {
      /*  make tree non-symmetric                        */

      if ( level == 3 && seq[level] > 7 ) /* 7 nodes for */
         break;                           /* level 3     */

      if ( level == 2 && seq[level] > 2 ) /* 2 nodes for */
         break;                           /* level 2     */

      /* allocate & init the node                        */

      if ( NOT (node = NEW(NODE_T)) )
      {
         printf("allocNode:  out of memory.\n");
         return(NULL);
      }
      node->seqNo = seq[level]++;   /* sequence number   */
      node->level = level;          /* tree level        */
      node->child = NULL;           /* child link        */
      node->next = NULL;            /* sibling link      */

      if ( NOT prevNode )           /* is it first one?  */
         topNode = node;            /* yes, set 1st node */
      else
         prevNode->next = node;     /* put node at end   */

      prevNode = node;              /* make 'node' prev  */

      /*  allocate level 2 children of level 1 nodes     */

      if ( level == 1 ) /* ask for 2 children for lvl 2  */
      {
         node->child = allocNode(2,level+1);
         continue;
      }

      /*  allocate level 3 children of level 2 nodes     */

      if ( level == 2 ) /* ask for 3 child nodes         */
         node->child = allocNode(3,level+1);
   }
   return( topNode );
}
/*=======================================================*/
/*
 *  placeNode - recursively set node coordinates and
 *              return next x coordinate. Y coordinate
 *              is incremented in recursive call and
 *              restored on return from recursion.
*/
/*=======================================================*/

int placeNode(NODE_T *topNode,int top,int left)
{
   NODE_T     *node;       /* the current node          */
   int        offset;      /* current left coordinate   */

   offset = left;
   node = topNode;

   while( node )     /* iterate until list is empty     */
   {
      /* set node bounding box corner coordinates       */

      node->x1 = offset;
      node->y1 = top;
      node->x2 = offset + HRAD + HRAD;
      node->y2 = top + VRAD + VRAD;

      /*
       * if the node is not a leaf, increment the top
       * coordinate and recursively place its children
       * starting at same x coordinate.
       * If it is a leaf node, just increment left
       * coordinate and place its sibling node.
      */

      if ( node->child )
         offset = placeNode(node->child,top+VCONS,offset);
      else
         offset += HCONS;

      node = node->next;   /*  set node to next in list  */
```

```
    }
    return( offset );       /*  return the left coordinate*/
}
/*=====================================================*/
/*
 *  freeNode - recursively deallocate nodes.
 *  This procedure iterates on a list of nodes.  If the
 *  node has child nodes, the procedure calls itself to
 *  free the child list before freeing the current node.
*/
/*=====================================================*/

void freeNode( NODE_T *topNode )
{
    NODE_T   *next;

    while( topNode )  /* stop when 'NULL'                  */
    {
        /*
         * if a node has children, recursively free all
         * child nodes before freeing self
        */

        if ( topNode->child )
            freeNode( topNode->child );

        next = topNode->next;   /* save link to sibling    */
        free( (void*)topNode ); /* free the node           */
        topNode = next;
    }
}
/*=====================================================*/
/*
 *  displayTree - recursively display entire tree.
 *  Procedure iterates on a list of nodes, displaying
 *  each node in the list.  If the node has a child list,
 *  the child link is drawn, then the procedure calls
 *  itself to display the child node list.  If the node
 *  has siblings, the sibling link is drawn and iteration
 *  continues with the node's sibling.
*/
/*=====================================================*/

void displayTree( NODE_T *topNode )
{
    NODE_T      *top;           /* local current node      */
    int         cx,cy;          /* geometric node center   */
    int         oldX,oldY;      /* previous node's edge    */

    top = topNode;

    while( top )      /* iterate until end of list (NULL) */
    {
        /*
         * find node center coordinates and save right,
         * center coordinates for drawing sibling link
        */

        cx = (top->x2 + top->x1)/2;
        cy = (top->y2 + top->y1)/2;
        oldX = top->x2;
        oldY = cy;
        displayNode( top,1,0 );   /* display the node */

        if ( top->child )
        {
            /*
             * draw a child link from bottom center of this
             * node, then recursively display child list
            */

            SetAPen(rp,2);
            Move( rp,cx,top->y2 );
            Draw( rp,cx,top->y2+VDIST );
            Move( rp,cx-4,top->y2+VDIST-4 );
            Draw( rp,cx,top->y2+VDIST );
            Draw( rp,cx+4,top->y2+VDIST-4 );

            displayTree( top->child );          /* recurse  */
        }
        if ( top = top->next )  /* set & test for sibling  */
        {
            /*
             * draw a sibling link from right center of last
             * node to left center of this node
            */

            SetAPen(rp,3);
            Move( rp,oldX,oldY );
            Draw( rp,top->x1,oldY);
            Move( rp,top->x1-4,oldY-4 );
            Draw( rp,top->x1,oldY );
            Draw( rp,top->x1-4,oldY+4 );
        }
    }
}
/*=====================================================*/
/*
 *  displayNode() - display a single node as an ellipse.
 *  This procedure is used for both regular display & for
 *  highlighting a node.
*/
/*=====================================================*/

void displayNode( NODE_T *node,int pn,int mode )
{
    char     levStr[10];    /* node level display string  */
    int      cx,cy;         /* node center coordinates    */

    if ( mode )             /* display in 'highlight' mode*/
        SetDrMd(rp,COMPLEMENT);
    else                    /* normal display             */
        SetDrMd(rp,JAM1);

    /*
     * get node center, draw an ellipse, then get the
     * node level and display it
     */

    cx = (node->x2 + node->x1)/2;
    cy = (node->y2 + node->y1)/2;
    SetAPen(rp,pn);
    DrawEllipse( rp,cx,cy,HRAD,VRAD);
    sprintf(levStr,"%d.%d",node->level,node->seqNo);
    Move( rp,cx-12,cy+3 );
    Text(rp,levStr,3);
    SetDrMd(rp,JAM1);       /* restore draw mode to normal */
}
/*=====================================================*/
/*
 *  handleInput() - monitor user input until end action
*/
/*=====================================================*/

void handleInput( NODE_T *tree )
{
    struct IntuiMessage  *msg;
    unsigned long        class;
    unsigned short       code;
    int                  endCode,x,y,menuNum;
    void                 (*search)();
    char                 *wMsg1,*wMsg2;

    search = NULL;
    endCode = 0;
    wMsg1 = "Pick a search method from the menu,";
    wMsg2 = "Then pick a node for the search.";
    displayMsg( 20,160,wMsg1,wMsg2,NULL,1 );

    for(;;)
    {
        Wait( 1 << window->UserPort->mp_SigBit );

        while( msg = (struct IntuiMessage *)
                                GetMsg(window->UserPort) )
        {
            class = msg->Class;  /* save message class      */
            code = msg->Code;    /* and message code.       */
            x = msg->MouseX;     /* and coordinates where   */
            y = msg->MouseY;     /* event occurred          */
            ReplyMsg( msg );     /* Reply right away.       */

            if ( class == CLOSEWINDOW )   /* close gadget   */
            {
                endCode = 1;
                break;
            }
            if ( class == MENUPICK && code != MENUNULL )
            {
                menuNum = MENUNUM(code); /* valid menu pick  */

                if ( menuNum == 0 )      /* 'QUIT' menu     */
                {
                    endCode = 1;
                    break;
                }
                switch( ITEMNUM(code) )
                {
                case 0:
                    search = doDFSearch;  /* set function ptr */
                    break;
                case 1:
                    search = doBFSearch;  /* ditto           */
                    break;
                default:
                    break;
                }
            }

            /*
             *  if we have valid function & select button was
             *  released, perform the search
            */

            if ( search &&
                 class == MOUSEBUTTONS && code == SELECTUP )
            {
                displayMsg( 20,160,wMsg1,wMsg2,NULL,0 );
                (*search)( tree,x,y );
                displayMsg( 20,160,wMsg1,wMsg2,NULL,1 );
            }
        }                       /*  end while              */
        if ( endCode )
            break;
    }                           /*  end for                */

    /*
     *  'end' event - jumped out of loop
     *  make sure message queue is cleared
    */
```

```
   while( msg = (struct IntuiMessage *)
                              GetMsg(window->UserPort) )
      ReplyMsg( msg );
}
/*=======================================================*/
/*
 *  waitForButton() - wait until user presses 'select'
 *                    button
*/
/*=======================================================*/

void waitForButton( void )
{
   struct IntuiMessage  *msg;
   unsigned long        class;
   unsigned long        code;

   for(;;)
   {
      Wait( 1 << window->UserPort->mp_SigBit );

      while( msg = (struct IntuiMessage *)
                              GetMsg(window->UserPort) )
      {
         class = msg->Class;
         code = msg->Code;
         ReplyMsg( msg );

         if ( class == MOUSEBUTTONS && code == SELECTUP )
            return;
      }
   }
}
/*=======================================================*/
/*
 * cleanUp()- free structures, close window, & close files
*/
/*=======================================================*/

void cleanUp( NODE_T *root, struct Menu *menu )
{
   /*
    *  if we have a menu and a window, clear the menu and
    *  recursively free entire menu strip
   */

   if ( menu )
   {
      if ( window )
         ClearMenuStrip( window );
      freeMenu( menu );
   }
   if ( window )  /* ditto                                */
      CloseWindow( window );

   /*    recursively free the entire tree                */

   if ( root )
      freeNode( root->child );

   if ( otp )    /* flush & close output file            */
   {
      fprintf(otp,"\nDone..\n");
      fflush(otp);
      fclose(otp);
   }
   closeLibs();
}
/*=======================================================*/
/*
 *  trace() - output indented trace messages for recursion
*/
/*=======================================================*/

void trace( msg,indent )
char  *msg;
int   indent;
{
   int   i;

   if ( NOT otp ) return;
   for ( i = 0; i < indent; i++ ) fprintf(otp," ");
   fprintf(otp,"%s\n",msg);
}
/*=======================================================*/
/*
 *  displayMsg() - Write or clear up to 3 lines of text
 *                 in a rastport
*/
/*=======================================================*/

void displayMsg( int x,int y,char *wMsg1,char *wMsg2,
                 char *wMsg3,int pen )
{
   SetAPen(rp,pen);

   Move(rp,x,y+textHeight);
   if ( wMsg1 )
     Text(rp,wMsg1,strlen(wMsg1));

   Move(rp,x,y+2*textHeight);
   if ( wMsg2 )
      Text(rp,wMsg2,strlen(wMsg2));

   Move(rp,x,y+3*textHeight);
   if ( wMsg3 )
      Text(rp,wMsg3,strlen(wMsg3));
}
/*=======================================================*/
/*
 *  openLibs() - open libraries
*/
/*=======================================================*/

int openLibs( void )
{
   IntuitionBase = (struct IntuitionBase *)OpenLibrary(
                        "intuition.library",INTUITION_REV);
   GfxBase = (struct GfxBase *)OpenLibrary(
                        "graphics.library",GRAPHICS_REV);

   return( GfxBase && IntuitionBase );
}
/*=======================================================*/
/*
 *  closeLibs() - close libraries
*/
/*=======================================================*/

void closeLibs( void )
{
   if ( IntuitionBase ) CloseLibrary(IntuitionBase);
   if ( GfxBase ) CloseLibrary(GfxBase);
}
/*=======================================================*/
/*
 *  displayWindow() - display an intuition window
*/
/*=======================================================*/

struct Window *displayWindow(int l,int t,int w,int h,
                             char *name)
{
   struct NewWindow  new;

   new.LeftEdge = l;
   new.TopEdge = t;
   new.Width = w;
   new.Height = h;
   new.DetailPen = -1;
```

```
   new.BlockPen = -1;
   new.Title = name;
   new.Flags = WINDOWCLOSE | SMART_REFRESH |
               WINDOWDRAG | ACTIVATE;
   new.IDCMPFlags = CLOSEWINDOW | MENUPICK | MOUSEBUTTONS;
   new.FirstGadget = NULL;
   new.CheckMark = NULL;
   new.Type = WBENCHSCREEN;
   return( (struct Window *)OpenWindow(&new) );
}
/*=====================================================*/
/*
 *  allocIText() - allocate & init an intuiText structure
*/
/*=====================================================*/

struct IntuiText *allocIText( short l,short t,
                              unsigned char *strPtr )
{
   struct IntuiText  *ts;

   if ( NOT (ts = NEW(struct IntuiText)) )
   {
      printf("\nallocIText:  out of memory.\n");
      return( NULL );
   }
   ts->FrontPen = 0;
   ts->BackPen = 1;
   ts->DrawMode = JAM1;
   ts->LeftEdge = l;
   ts->TopEdge = t;
   ts->ITextFont = NULL;
   ts->IText = strPtr;
   ts->NextText = NULL;
   return( ts );
}
/*=====================================================*/
/*
 *  allocMenuItem() - allocate & init a menu item structure
*/
/*=====================================================*/

struct MenuItem *allocMenuItem( short l,short t,
                                short w,short h,
                                struct IntuiText *strPtr )
{
   struct MenuItem   *mi;

   if ( NOT (mi = NEW(struct MenuItem)) )
   {
      printf("\nallocMenuItem:  out of memory.\n");
      return( NULL );
   }
   mi->NextItem = NULL;
   mi->LeftEdge = l;
   mi->TopEdge = t;
   mi->Width = w;
   mi->Height = h;
   mi->Flags = ITEMTEXT | ITEMENABLED | HIGHCOMP;
   mi->MutualExclude = NULL;
   mi->ItemFill = (APTR)strPtr;
   mi->Command = NULL;
   mi->SubItem = NULL;
   mi->SelectFill = NULL;
   return( mi );
}
/*=====================================================*/
/*
 *  allocMenu() - allocate & init a menu structure
*/
/*=====================================================*/

struct Menu *allocMenu( short l,short t,
                        short w,short h,
                        unsigned char *strPtr )
{
   struct Menu   *menu;

   if ( NOT (menu = NEW(struct Menu)) )
   {
      printf("\nallocMenu:  out of memory.\n");
      return( NULL );
   }
   menu->NextMenu = NULL;
   menu->LeftEdge = l;
   menu->TopEdge = t;
   menu->Width = w;
   menu->Height = h;
   menu->Flags = MENUENABLED;
   menu->MenuName = strPtr;
   menu->FirstItem = NULL;
   return( menu );
}
/*=====================================================*/
/*
 *  initMenu() - allocate & initialize all menus
*/
/*=====================================================*/

struct Menu *initMenu( void )
{
   struct Menu       *topMenu,*lastMenu;
   struct MenuItem   *topItem,*lastItem;
   struct IntuiText  *text;
   short             mWidth,mHeight,temp;

   /* set width & height & init first menu list          */

   mWidth = 100;  mHeight = textHeight+2;
   topMenu = allocMenu(0,0,mWidth,mHeight,"PROJECT");
   text = allocIText(10,0,"Quit");

   if ( topMenu == NULL || text == NULL )
      return( NULL );      /* didn't get it              */

   topMenu->FirstItem = allocMenuItem(0,0,
                                      mWidth,mHeight,text);

   /* allocate next menu list & link the menu lists      */

   temp = mWidth + 5;
   topMenu->NextMenu = lastMenu =
          allocMenu(temp,0,mWidth,mHeight,"PERFORM");

   if ( NOT topMenu->FirstItem || NOT lastMenu )
   {                          /* didn't get it           */
      freeMenu( topMenu ); /* free the first & return    */
      return( NULL );
   }

   /* allocate & link together the subitems for 2d menu  */

   text = allocIText(10,0,"DF Search");
   topItem = lastItem = allocMenuItem( 0,0,
                                      mWidth,mHeight,text);

   if ( NOT text || NOT lastItem )
   {                                 /* didn't get them   */
      freeMenu( topMenu );
      return( NULL );
   }
   lastMenu->FirstItem = topItem;   /* link the items     */
   text = allocIText(10,0,"BF Search");
   lastItem->NextItem = allocMenuItem( 0,mHeight,
                                       mWidth,mHeight,text);
   lastItem = lastItem->NextItem;

   if ( NOT text || NOT lastItem )  /* didn't get 'em     */
   {
      freeMenu( topMenu );
      return( NULL );
   }
   return( topMenu );
}
/*=====================================================*/
/*
 *  freeMenu - free entire menu structure
*/
/*=====================================================*/

void freeMenu( struct Menu *topMenu )
{
   struct Menu       *nextMenu;
   struct MenuItem   *menuItem,*nextItem;
   struct IntuiText  *text;

  while( topMenu )     /* until end of menu list         */
   {
      if ( menuItem = topMenu->FirstItem )
      {
         while( menuItem )    /* until end of item list  */
         {
            if (text=(struct IntuiText*)menuItem->ItemFill)
               free(text);    /* free IntuiText          */

            /* free this menuItem & set pointer to next  */

            nextItem = menuItem->NextItem;
            free( menuItem );
            menuItem = nextItem;
         }
      }

      /* now free the menu list ptr & set ptr to next    */

      nextMenu = topMenu->NextMenu;
      free( topMenu );
      topMenu = nextMenu;
   }
}
```

•AC•

The AMICUS

Public Domain Software Library

This software is collected from user groups and electronic bulletin boards around the nation. Each Amicus disk is nearly full, and is fully accessible from the Workbench. If source code is provided for any program, then the executable version is also present. This means that you don't need the C compiler to run these programs. An exception is granted for those programs only of use to people who own a C compiler.

Note: Each description line below may include something like 'S-O-E-D', which stands for 'source, object file, executable and documentation'. Any combination of these letters indicates what forms of the program are present. Basic programs are presented entirely in source code format.

AMICUS Disk 1
ABasic programs: Graphics
3DSolids — 3d solids modeling prog. w/sample data files
Blocks — draws blocks
Cubes — draws cubes
Durer — draws pictures in the style of Durer
FScape — draws fractal landscapes
Hidden — 3D drawing program, w/ hidden line removal
JPad — simple paint program
Optical — draw several optical illusions
PaintBox — simple paint program
Shuttle — draws the Shuttle in 3d wireframe
SpaceArt — graphics demo
Speaker — speech utility
Sphere — draws spheres
Spiral — draws color spirals
ThreeDee — 3d function plots
Topography — artificial topography
Wheels — draws circle graphics
Xenos — draws fractal planet landscapes
ABasic programs: Tools
AddressBook — simple database program for addresses
CardFile — simple card file database program
Demo — multiwindow demo
KeyCodes — shows keycodes for a key you press
Menu — run many ABasic programs from a menu
MoreColors — way to get more colors on the screen at once, using aliasing
shapes — simple color shape designer SpeakIt speech and narrator demo
ABasic programs: Games
BrickOut — classic computer brick wall game
Othello — also known as 'go'
Saucer — simple shoot-em-up game
Spelling — simple talking spelling game
ToyBox — selectable graphics demo
ABasic programs: Sounds
Entertainer — plays that tune
HAL9000 — pretends it's a real computer
Police — simple police siren sound
SugarPlum — plays "The Dance of the Sugarplum Fairies"
C programs:
ATerm — simple terminal program, S-E
cc — aid to compiling with Lattice C
decvnt — opposite of CONVERT for cross developers
Dotty — source code to the 'dotty' window demo
echox — unix-style filename expansion, partial S,O-D
fasterlp — explains use of fast-floating point math
FixDate — fixes future dates on all files on a disk, S-E
freedraw — simple Workbench drawing prog.,S-E
GfxMem — graphic memory usage indicator, S-E
Grep — searches for a given string in a file with docs.
ham shows off the hold-and-modify method of color generation
IBM2Amiga — fast parallel cable transfers between an IBM and an Amiga
Mandel — Mandelbrot set program, S-E
moire — patterned graphic demo, S-E
objfix — makes Lattice C object file symbols visible to Wack, S-E
quick — quick sort strings routine
raw — example sample window I/O
setlace — turns on interlace mode, S-E
sparks — qix-type graphic demo, S-E
Other executable programs:
SpeechToy — speech demonstration
WhichFont — displays all available fonts
Texts:
68020 — describes 68020 speedup board from CSA
Aliases — explains uses of the ASSIGN command
Bugs — known bug list in Lattice C 3.02
CLICard — reference card for AmigaDOS CLI
CLICommands — guide to using the CLI
Commands — shorter guide to AmigaDOS CLI commands
EdCommands — guide to the ED editor
Filenames — AmigaDOS filename wildcard conventions
HalfBright — explains rare graphics chips that can do more colors
ModemPins — description of the serial port pinout
RAMdisks — tips on setting up your RAM: disk
ROMWack — tips on using ROMWack
Sounds — explanation of Instrument demo sound file format
Speed — refutation of Amiga's CPU and custom chip speed
WackCmds — tips on using Wack

AMICUS Disk 2
C programs:
alib — AmigaDOS object library manager , S-E
ar — text file archive program, S-E
fixobj — auto-chops executable files
shell — simple CLI shell, S-E
sq, usq — file compression programs, S-E
YachtC — a familiar game, S-E
Make — a simple 'make' programming utility, S-E
Emacs — an early version of the Amiga text editor, S-E-D
Assembler programs:
bsearch.asm — binary search code
qsort.asm — Unix compatible qsort() function, source and C test program
setjmp.asm — setjmp() code for Lattice 3.02
SVprintf — Unix system V compatible printf()
trees.o — Unix compatible tree() function, O-D
(This disk formerly had IFF specification files and examples. Since this spec is constantly updated, the IFF spec files have been moved to their own disk in the AMICUS collection.)
John Draper Amiga Tutorials:
Animate — describes animation algorithms
Gadgets — tutorial on gadgets
Menus — learn about intuition menus

AMICUS Disk 3
C programs:
Xref — a C cross-reference gen., S-E
6bitcolor — extra-half-bright chip gfx demo, S-E
Chop — truncate (chop) files down to size, S-E
Cleanup — removes strange characters from text files
CR2LF — converts carriage returns to line feeds in Amiga files, S-E
Error — adds compile errors to a C file, S
Hello — window ex. from the RKM, S
Kermit — generic Kermit implementation, flakey, no terminal mode, S-E
Scales — sound demo plays scales, S-E
SkewB — Rubik cube demo in hi-res colors, S-E
AmigaBasicProgs(dir)
Automata — cellular automata simulation
CrazyEights — card game
Graph — function graphing programs
WitchingHour — a game
ABasic programs:
Casino — games of poker, blackjack, dice, and craps
Gomoku — also known as 'othello'
Sabotage — sort of an adventure game
Executable programs:
Disassem — a 68000 disassembler, E-D
DpSlide — shows a given set of IFF pictures, E-D
Arrange — a text formatting program, E-D
Assembler programs:
Argoterm — terminal program with speech and Xmodem, S-E

AMICUS Disk 4 Files from the original Amiga Technical BBS
Note that some of these files are old, and refer to older versions of the operating system. These files came from the Sun system that served as Amiga technical support HQ for most of 1985. These files do not carry a warranty, and are for educational purposes only. Of course, that's not to say they don't work.

Complete and nearly up-to-date C source to 'image.ed', an early version of the Icon Editor. This is a little flaky, but compiles and runs.

An Intuition demo, in full C source, including files: demomenu.c, demomenu2.c, demoreq.c, getascii.c, idemo.c, idemo.guide, idemo.make, idemoall.h, nodos.c, and txwrite.c
addmem.c — add external memory to the system
bobtest.c — example of BOB use
consoleIO.c — console IO example
creaport.c — create and delete ports
creastdi.c — create standard I/O requests
creatask.c — creating task examples
diskio.c — example of track read and write
dotty.c — source to the 'dotty window' demo
dualplay.c — dual playfield example
flood.c — flood fill example
freemap.c — old version of 'freemap'
geltools.c — tools for VSprites and BOBs
gfxmem.c — graphic memory usage indicator
hello.c — window example from RKM
inputdev.c — adding an input handler to the input stream
joystik.c — reading the joystick
keybd.c — direct keyboard reading
layertes.c — layers examples
mousport.c — test mouse port
ownlib.c, ownlib.asm — example of making your own library with Lattice
paratest.c — tests parallel port commands
seritest.c — tests serial port commands
serisamp.c — example of serial port use
prinintr.c — sample printer interface code
prtbase.h — printer device definitions
regintes.c — region test program
setlace.c — source to interlace on/off program
setparallel.c — set the attributes of the parallel port
SetSerial.c — set the attributes (parity, data bits) of the serial port
singplay.c — single playfield example
speechtoy.c — source to narrator and phonetics demo
timedely.c — simple timer demo
timer.c — exec support timer functions
timrstuf.c — more exec support timer functions
WhichFont.c — loads and displays all available system fonts
process.i and prtbase.i assmebler include files:
autorqstr.txt — warnings of deadlocks with autorequesters
consoleIO.txt — copy of the RKM console I/O chapter
diskfont.txt — warning of disk font loading bug
fullfunc.txt — list of #defines, macros, functions
inputdev.txt — preliminary copy of the input device chapter
License information on Workbench distribution license
printer pre-release copy of the chapter on printer drivers, from RKM 1.1
v11fd.txt 'diff' of .fd file changes from version 1.0 to 1.1 v28v1.diff 'diff' of include file changes from version 28 to 1.0

AMICUS Disk 5 Files from the Amiga Link / Amiga Information Network
Note that some of these files are old, and refer to older versions of the operating system. These files are from Amiga Link. For a time, Commodore supported Amiga Link, aka AIN, for online developer technical support. It was only up and running for several weeks. These files do not carry a warranty, and are for educational purposes only. Of course, that's not to say they don't work.
A demo of Intuition menus called 'menudemo', in C source
whereis.c — find a file searching all subdirectories
bobtest.c — BOB programming example
sweep.c — sound synthesis example
Assembler files:
mydev.asm — sample device driver
mylib.asm — sample library example
mylib.i
mydev.i
asmsupp.i
macros.i — assembler include files
Texts:
amigatricks — tips on CLI commands
extdisk — external disk specification
gameport — game port spec
parallel — parallel port spec
serial — serial port spec
v1.1update — list of new features in version 1.1
v1.1h.txt — 'diff' of include file changes from version 1.0 to 1.1
Files for building your own printer drivers, including dospecial.c, epsondata.c, init.asm, printer.c, printer.link, printertag.asm, render.c, and wait.asm. This disk does contain a number of files describing the IFF specification. These are not the latest and greatest files, but remain here for historical purposes. They include text files and C source examples. The latest IFF spec is elsewhere in this library.

AMICUS Disk 6 IFF Pictures
This disk includes the DPSlide program, which can view a given series of IFF pictures, and the 'showpic' program, which can view each file at the click of an icon. The pictures include a screen from ArticFox, a Degas dancer, the guys at Electronic Arts, a gorilla, horses, King Tut, a lighthouse, a screen from Marble Madness, the Bugs Bunny Martian, a still from an old movie, the Dire Straits moving company, a screen from Pinball Contruction Set, a TV newcaster, the PaintCan, a world map, a Porsche, a shuttle mission patch, a tyrannosaurus rex, a planet view, a VISA card, and a ten-speed.

AMICUS Disk 7 DigiView HAM demo picture disk
This disk has pictures from the DigiView hold-and-modify video digitizer. It includes the ladies with pencils and lollypops, the young girl, the bulldozer, the horse and buggy, the Byte cover, the dictionary page, the robot and Robert. This includes a program to view each picture separately, and all together as separate, slidable screens. The 'seeilbm' program, to turn any screen into an IFF picture.

AMICUS Disk 8
C programs:
Browse — view text files on a disk, using menus S-E-D
Crunch — removes comments and white space from C files, S-E
IconExec — EXECUTE a series of commands from Workbench S-E
PDScreen Dump — dumps Rastport of highest screen to printer
SetAlternate — sets a second image for an icon, when clicked once S-E
SetWindow — makes windows for a CLI program to run under Workbench S-E
SmallClock — a small digital clockin a window menu bar
Scrimper — the screen printer in the fourth AC S-E
Amiga Basic Programs:
(Note: Many of these programs are present on AMICUS Disk 1. Several of these were converted to Amiga Basic, & included here.)
AddressBook — a simple address book database
Ball — draws a ball
Cload — program to convert Compuserve hex files to binary, S-D
Clue — the game, Intuition driven
ColorArt — art drawing program
DeluxeDraw — the drawing program in the 3rd AC, S-D
Eliza — conversational computer psychologist
Othello — the game, as known as 'go'
RatMaze — 3D ratmaze game
ROR — boggling graphics demo
Shuttle — draws 3D pictures of the space shuttle
Spelling — simple spelling program
YoYo — wierd zero-gravity yo-yo demo, tracks yo-yo to the mouse
Executable programs:
3Dcube — Modula-2 demo of a rotating cube
AltIcon — sets a second icon image, displayed when the icon is clicked
AmigaSpell — a slow but simple spell checker, E-D
arc — the ARC file compression prog must for telecom, E-D
Bertrand — graphics demo
disksalvage — prog. to rescue trashed disks, E-D
KwikCopy — a quick but nasty disk copy program: ignores errors, E-D
LibDir — lists hunks in an object file E-D
SaveILBM — saves any screen as IFF pic.E-D ?? ScreenDump shareware screen dump prog, E only
StarTerm — version 2.0, term program, XmodemE-D
Texts:
LatticeMain — tips on fixing _main.c in Lattice
GDiskDrive — make your own 5 1/4 drive
GuruMed — explains the Guru numbers
Lat3.03bugs — bug list of Lattice C version 3.03
MForgeRev — user's view of the MicroForge HD
PrintSpooler — EXECUTE-based print spool prog.
.BMAP files:
These are the necessary links between Amiga Basic and the system libraries. To take advantage of the Amiga's capabilities in Basic, you need these files. BMAPs are included for 'clist', 'console', 'diskfont', 'exec', 'icon', 'intuition', 'layers', 'mathtfp', mathieeedoubas', 'mathieeesingbas', 'mathtrans', 'potgo', 'timer' and 'translator'.

AMICUS Disk 9
Amiga Basic Programs:
FlghtSim — simple flight simulator program
HuePalette — explains Hue, Saturation, & Intensity
Requester — ex. of requesters from Amiga Basic
ScrollDemo — demonstrates scrolling capabilities
Synthesizer — sound program
WorldMap — draws a map of the world
Executable programs:
Boing! — latest Boing! demo,with selectable speed,E
Brush2C — converts an IFF brush to C data instructions, initialization code, E
Brush2Icon — converts IFF brush to an icon, E
Dazzle — graphics demo, tracks to mouse, E
DeciGEL — assembler program for stopping 68010 errors, S-E-D
Klock — menu-bar clock and date display, E
life — the game of life, E
TimeSet — Intuition-based way to set the time & date
EMEmacs — another Emacs, more oriented to word processing, S-E-D
MyCLI — a CLI shell, works without the Workbench, S-E-D
Texts:
FnctnKeys — read function keys from Amiga Basic
HackerSIn — explains how to win the game 'hacker'
Ist68010 — guide to installing a 68010 in your Amiga
PrinterTip — sending escape sequences to your printer
StartupTip — tips on setting up your startup-sequence file
XfrmrReview — list of Transformer programs that work
Printer Drivers:
Printer drivers for the Canon PJ-1080A, the C Itoh Prowriter, an improved Epson driver that eliminates streaking, the Epson LQ-800, the Gemini Star-10, the NEC 8025A, the Okidata ML-92, the Panasonic KX-P10xx family, and the Smith-Corona D300, with a document describing the installation process.

AMICUS Disk 10 Instrument sound demos
This is an icon-driven demo, circulated to many dealers. It includes the sounds of an acoustic guitar, an alarm, a banjo, a bass guitar, a boink, a calliope, a car horn, claves, water drip, electric guitar, a flute, a harp arpegio, a kickdrum, a marimba, a organ minor chord, people talking, pigs, a pipe organ, a Rhodes piano, a saxophone, a sitar, a snare drum, a steel drum, bells, a vibrophone, a violin, a wailing guitar, a horse whinny, and a whistle.

AMICUS Disk 11
C programs
dirutil — Intuition-based, CLI replacement manager S-E
cpri — shows and adjusts priority of CLI processes, S-E
ps — shows info on CLI processes, S-E
vidtex — displays Compuserve RLE pics, S-E
AmigaBasic programs
pointered — pointer and sprite editor program
optimize — optimization ex ample from AC article
calendar — large, animated calendar, diary and date book program
amortize — loan amortizations
brushtoBOB — converts small IFF brushes to AmigaBasic BOB OBJECTS
grids — draw and play waveforms
hilbert — draws Hilbert curves
madlib — mad lib story generator
mailtalk — talking mailing list program
meadows3D — 3D graphics program, from A C™ article
mousetrack — mouse tracking example in hires mode
slot — slot machine game
tictactoe — the game
switch — pachinko-like game
weird — makes strange sounds
Executable programs
cp — unix-like copy command, E
cls — screen clear, S-E
diff — unix-like stream editor uses 'diff' output to fix files
pm — chart recorder performances indicator
Assembler programs
cls — screen clear and CLI arguments example
Modula-2
trails — moving-worm graphics demo
caseconvert — converts Modula-2 keywords to uppercase
Forth
Breshehan circle algorithm example
Analyze — 12 templates for the spreadsheet Analyze
There are four programs here that read Commodore 64 picture files. They can translate Koala Pad, Doodle, Print Shop and News Room graphics to IFF format. Getting the files from your C-64 to your Amiga is the hard part.

AMICUS Disk 12
Executable programs
blink — 'alink' compatible linker, but faster, E-D
clean — spins the disk for disk cleaners, E-D
epsonset — sends Epson settings to PAR from menu E-D
showbig — view hi-res pics in low-res superbitmap, E-D
speaktime — tell the time, E-D
undelete — undeletes a file, E-D
cnvapldhm — converts Apple][low, medium and high res pictures to IFF, E-D
menued — menu editor produces C code for menus, E-D
quick — quick disk-to-disk nibble copier, E-D
quickEA — copies Electronic Arts disks, removes protection, E-D
txed 1.3 — demo of text editor from Micrositths,E-D
C programs
spin3 — rotating blocks graphics demo, S-E-D
popcli — start a new CLI at the press of a button, like Sidekick, S-E-D
vsprite — VSprite example code from Commodore, S-E-D
AmigaBBS — Amiga Basic bulletin board prog., S-D
Assembler programs
star10 — makes star fields like Star Trek intro,S-E-D
Pictures
Mount Mandelbrot — 3D view of Mandelbrot set
Star Destroyer — hi-res Star Wars starship
Robot — robot arm grabbing a cylinder
Texts
vendors — Amiga vendors, names, addresses
cardco — fixes to early Cardco memory boards
cinclude — cross-reference to C include files
mindwalker — clues to playing the game well
slideshow — make your own slideshows from the Kaleidoscope disk

AMICUS Disk 13
Amiga Basic programs
Routines from Carolyn Scheppner of CBM Tech Support, to read and display IFF pictures from Amiga Basic. With documentation. Also included is a program to do screen prints in Amiga Basic, and the newest BMAP files, with a corrected ConvertFD program. With example pictures, and the SaveILBM screen capture program.

Routines to load and play FutureSound and IFF sound files from Amiga Basic, by John Foust for Applied Visions. With documentation and C and assembler source for writing your own libraries, and interfacing C to assembler in libraries. With example sound.

Executable programs
gravity — Sci Amer Jan 86 gravitation graphic simulation, S-E-D
Texts
MIDI — make your own MIDI instrument interface, documentation & a hi-res schematic.

AMICUS Disk 14
Several programs from Amazing Computing issues:
Tools
Dan Kary's — C structure index program, S-E-D
Amiga Basic programs:
BMAP Reader — by Tim Jones
IFFBrush2BOB — by Mike Swinger
AutoRequester — example
DOSHelper — Windowed help system for CLI commands, S-E-D
PETrans — translates PET ASCII files to ASCII files, S-E-D
C Squared — Graphics program from Scientific American, Sept 86, S-E-D
crlf — adds or removes carriage returns from files, S-E-D
dpdecode — decrypts Deluxe Paint, remo
ves copy — protection, E-D
queryWB — asks Yes or No from the user returns exitt code, S-E
vc — VisiCalc type spreadsheet, no mouse control, E-D
view — views text files with window and
slider — gadget, E-D
Oing, Sproing, yaBoing, Zoing are sprite-based Boing! style — demos, S-E-D
CLIClock, sClock, wClock are window border clocks, S-E-D
Texts
An article on long-persistance phospor monitors, tips on making brushes of odd shapes in Deluxe Paint, and recommendations on icon interfaces from Commodore-Amiga.

AMICUS Disk 15
The C programs include:
'pr' — a file printing utility, which can print files in the background, and with line numbers and control character filtering.
'fm' — displays a chart of the blocks allocated on a disk.
'Ask' — questions an 'execute' file, returns an error code to control the execution in that batch file
'Stat' — an enhanced version of AmigaDOS 'status' command.
'Dissolve' — random-dot dissolve demo displays IFF picture slowly, dot by dot, in a random fashion.
'PopCLI2' — invoke new CLI window at the press of a key.
The executable programs include:
'Form' — file formatting program through the printer driver to select print styles
'DiskCat' — catalogs disks, maintains, sorts,merges lists of disk files
'PSound' — SunRize Industries' sampled sound editor & recorder
'Iconmaker' — makes icons for most programs
'Fractals' — draws great fractal seascapes and mountain scapes.
'3D Breakout' — 3D glasses, create breakout in a new dimension
'AmigaMonitor' — displays lists of open files,
memory use, — tasks, devices and ports in use.
'Cosmoroids' — version of 'asteroids' for the Amiga.
'Sizzlers' — high resolution graphics demo written in Modula 2.

Texts:
'ansi.txt' — explains escape sequences the CON: device responds to.
'FKey' — includes template for making paper to sit in the tray at the top of the Amiga keyboard.
'Spawn' — programmer's document from Commodore Amiga, describs ways to use the Amiga's multitasking capabilities in your own programs.
AmigaBasic programs:
'Grids' — draw sound waveforms, and hear them played.
'Light' — a version of the Tron light-cycle video game.
'MigaSol' — a game of solitaire.
'Stats' — program to calculate batting averages
'Money' — "try to grab all the bags of money that you can."
AMICUS 15 also includes two beautiful IFF pictures, of the enemy walkers from the ice planet in Star Wars, and a picture of a cheetah.

AMICUS Disk 16
'juggler' — demo by Eric Graham. a robot juggler bouncing three mirrored balls, with sound effects. Twenty-four frames of HAM animation are flipped quickly to produce this image. You control the speed of the juggling. The author's documentation hints that this program might someday be available as a product.
IFF pictures
parodies of the covers of Amiga World and Amazing Computing.
C programs:
'Inputhandler' — example of making an input handler.
'FileZap3' — binary file editing program
'ShowPrint' — displays IFF picture, and prints it.
'Gen' — program indexes and retrieves C structures and variables declared in the Amiga include file system.
Executable Programs:
'FixHunk2' — repairs an executable program file for expanded memory
'ms2smus' — converts Music Studio files to IFF standard 'SMUS' format. I have heard this program might have a few bugs, especially in regards to very long songs, but it works in most cases.
'Missile' — Amiga version of the 'Missile Command' video game,

This disk also contains several files of scenarios for Amiga Flight Simulator II. By putting one of these seven files on a blank disk, and inserting it in the drive after performing a special command in this game, a number of interesting locations are preset into the Flight Simulator program. For example, one scenario places your plane on Alcatraz, while another puts you in Central Park

AMICUS Disk 17
Telcommunications disk whuch contains six terminal programs.
"Comm" V1.33 — term prog. with Xmodem, WXmodem,
"ATerm" V7.2 — term prog. includes Super Kermit
"VT-100"V2.6 — Dave Wecker's VT-100 emulator with Xmodem,Kermit, and scripting
"Amiga Kermit" — V4D(060) port of the Unix C-Kermit
"VTek" V2.3.1 — Tektronix graphics terminal emulator based on the VT-100 prog. V2.3 and contains latest 'arc' file compression
"AmigaHost" — V0.9 for Compuserve. Includes RLE graphics abilities & CIS-B file transfer protocol.
"FixHunk" — expansion memory necessity
"FixObj" — removes garbage characters from modem received files
"Txt" — filters text files from other systems to be read by the Amiga E.C.
"addmem" — executeable version for use with mem expansion article in AC v2.1
'arc' — file documentation and a basic tutorial on un 'arc'ing files
"arcre" — for makeing "arc" files E.C.

AMICUS Disk 18
Logo — Amiga version of the popular computer language, with example programs, E-D
TV*Text — Demo version of the TV*Text character generator

PageSetter — Freely distributable versions of the updated PagePrint and PageIFF programs for the PageSetter desktop publishing package.
FullWindow — Resizes any CLI window using only CLI commands, E-D
Life3d — 3-D version of Conway's LIFE program, E-D
Defdisk — CLI utility to re-assign a new Workbench disk, S-E-D
Calendar.WKS — Lotus-compatible worksheet that makes calendars
SetKey — Demo of keyboard key re-programmer, with IFF picture to make function key labels, E-D
VPG — Video pattern generator for aligning monitors, E-D
HP-10C — Hewlett-Packard-like calculator, E-D
SetPrefs — Change the Preferences settings on the fly, in C, S-E-D
StarProbe — Program studies stellar evolution. C source included for Amiga and MS-DOS, S-E-D
ROT — C version of Colin French's AmigaBasic ROT program from Amazing Computing. ROT edits and displays polygons to create three dimensional objects. Up to 24 frames of animation can be created and displayed. E-D
Scat — Like Ing, windows on screen run away from the mouse, E-D
DK — Decays* the CLI window into dust, in Modula 2, S-E-D
DropShadow2 — Adds layered shadows to Workbench windows, E-D

AMICUS Disk 19
This disk carries several programs from Amazing Computing. The IFF pictures on this disk include the Amiga Wake part T-shirt logo, a sixteen-color hi-res image of Andy Griffith, and five Amiga Live! pictures from the Amazing Stories episode that featured the Amiga.
Solve — Linear equation solver in assembly language, S-E-D
Gadgets — Bryan Catley's AmigaBasic tutorial, S-D
Household — Bryan Catley's AmigaBasic household inventory program, S-D
Waveform — Jim Shields' Waveform Workshop: AmigaBasic, S-D
DiskLib — John Kennan's AmigaBasic disk librarian program, S-D
Subscripts — Ivan Smith's AmigaBasic subscript example, S-D
String, Boolean — C programs and executables for Harriet Maybeck Tolly's Intuition tutorials, S-E-D
Skinny C — Bob Riemersma's example for making small C programs, S-E-D
COMAL.h — Make C look like COMAL header file, S-D
EmacsKey — Makes Emacs function key definitions by Greg Douglas, S-D
AMon 1.1 — Snoop on system resource use, E-D
BTE — Bard's Tale character editor, E-D
Size — CLI program shows the size of a given set of files, E-D
WinSize — CLI window utility resizes current window, S-E-D

AMICUS Disk 20
Compactor, Decoder Steve Michel AmigaBasic tools, S-D
BobEd — BOB and sprite editor written in C, S-E-D
SpriteMasterII — Sprite editor and animator by Brad Kieler, E-D
BlitLab — Blitter chip exploration C program by Tomas Rokicki, S-E-D
FPic — Image processing program by Bob Bush loads and saves IFF images, changes them with several techniques, E-D
Bankn — Complete home banking prog., balance your checkbook! E-D

AMICUS Disk 21
Target — Makes each mouse click sound like a gunshot, S-E-D
Sand — Simple game of sand that follows the mouse pointer, E-D
PropGadget — Harriet Maybeck Tolly's proportional gadget example, S-E
EHB — Checks to see if you have extra-half-bright graphics, S-E-D
Piano — Simple piano sound program
CelScripts — Makes cel animation scripts for Aegis Animator, in AmigaBasic

This disk has electronic catalogs for AMICUS disks 1 to 20 and Fish disks 1 to 80. They are viewed with the DiskCat program, included here.

AMICUS Disk 22
Cycles — Light cycle game, E-D
Show_PrintII — Views and prints IFF pictures, including larger than screen
PrtDrvGen2.3 — Latest version of a printer driver generator
Animations — VideoScape animations of planes and boing ball
Garden — Makes fractal gardenscapes
BasicSorts — Examples of binary search and insertion sort in AmigaBasic

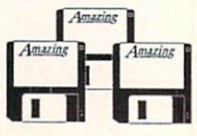
AMICUS Disk 23
An AMICUS disk completely dedicated to music on the Amiga. This disk contains two music players, songs, instruments, and players to bring the thrill of playing "Big Sound" on your Amiga
Instruments — a collection of 25 instruments for playing and creating music. The collection ranges from Cannon to Marimba
List INSTR — program to list the instruments DMCS will not load as well as list the origins for any instrument.
Music — a collection of 14 Classical pieces
1812Overture — The 16 minute classical feature complete with Cannon!
Three Amiga Music Players:
SMUSPlay
MusicCraft2SMUS
MusicStudio2SMUS

AMICUS Disk 24
Sectorama — A disk sector editor for any AmigaDOS file-structured device, recover files from a trashed hard disk. By David Joiner of Microillusions
Iconize — Reduces the size of IFF images, companion program, Recolor, remaps the palette colors of one picture to use the palette colors of another. Using these programs and a tool to convert IFF brushes to Workbench icons, make icons look like miniatures of the pictures.
CodeDemo — Modula-2 program converts assembler object files to inline CODE statements. Comes with a screen scrolling example
AmiBug — Workbench hack makes the same fly walk across the screen at random intervals. Otherwise, completely harmless.
BNTools — Three examples of assembly language code from Bryce Nesbitt:
1. SetLace.prog to switch interlace on&off.
2. Why, replace AmigaDOS CLI Why
3. Loadit, prog to load a file into memory until a reboot. (Only the most esoteric hackers will find Loadit useful.)
Monolace — CLI program resets Preferences to several colors of monochrome & interlace screens. C source is included, works with DisplayPref, a CLI program which displays the current Preferences settings.
BoingMachine — A ray-traced animation of a perpetual motion Boing-making machine, includes the latest version of the Movie program, which has the ability to play sounds along with the animation. By Ken Offer
Daisy — Example of using the translator and narrator devices to make the Amiga talk. It is written in C.
QuickFlix — Script-driven animation and slideshow program flips through IFF images.

BMon — System monitor AmigaBasic program ; perform simple manipulations of memory.
Moose — Random background program, a small window opens with a moose resembleing Bullwinklesaying witty phrases user definable.
DGCS — Deluxe Grocery Construction Set, simple Intuition-based prog for assembling and printing a grocery list.
The Virus Check directory holds several programs relating to the software virus that came to the US from pirates in Europe as detailed in Amazing Computing V2.12. Bill Koester's full explanation of the virus code is included. One program checks for the software virus on a Workbench disk; the second program checks for the virus in memory, which could infect other disks.

AMICUS Disk 25
Nemesis — Graphics demo pans through space towards the mythical dark twin of the sun with wonderful music and space graphics.
The KickPlay directory holds text that describes several patches to the Kickstart disk. For Amiga 1000 hackers who feel comfortable patching a disk in hexadecimal, KickPlay offers the chance to automatically do an ADDMEM for old expansion memory, as well as the ability to change the picture of the "insert Workbench" hand. A program is also included for restoring the correct checksum of the Kickstart disk.
KeyBird — BASIC prog edits keymaps, adjust the Workbench keymaps or create your own.
8ColorWB — Modifies the Workbench so three bitplanes are used, icons can have eight colors, instead of four, eight-color icons are included. Public domain program "zapicon" or "brush2icon" converts eight-color IFF brushes to icons, to use Deluxe Paint to make icons for this new Workbench.
BrushIcon — Converts brushes to icons (bizarr docs).
Egraph — Graphing prog reads [x,y] values from a file and displays them on the screen, similar to the same-named Unix program.
Keep 1.1 — Message-managing program for telecommunications, lets you save messages from an online transcript to another file, understands the message format of the national networks and several types of bulletin board software. Moves through the transcript and save messages
Kill.fastdir — Speed up directory access, it creates a small file in each directory on a disk which contains the infor-mation about the files, will also remove all the "fastdir" files from each directory. by CLImate's authors
The LaceWB program changes between interlace and non-interlace Workbench. Previously, you were forced to reboot after changing Preferences to an interlaced screen. This program flips between the normal and extended screen heights.
PW_Utility — A shareware utility for ProWrite users, changes margin settings and font types.
Guru — A CLI program, prints out probable causes for Guru mediations; C source included.
DiskWipe — Latest from Software Distillery, removes files from directories or disk drives, much faster than "delete."
Snow — AmigaBasic makes snowflake designs.
Mlist — Mailing list database.
Softballstats — Maintain softball statistics/ team records.
Dodge — Short Modula-2 program moves the Workbench screen around after a period of time, prevents monitor burn-in.

AMICUS Disk 26
Todor Fay's SoundScape module code from his Amazing Computing articles. The source to Echo, Chord, TX, and VU is included. The Lattice and Manx C source code is here, along with the executable modules.
Ciaz2 — Update of prog to convert IFF images to PostScript files for printing on laser printers
SDBackup — Hard disk backup prog with Lempel-Ziv compression to reduce the necessary number of disks.
TCB — Prints information about tasks and processes in the system; assembler source is included.
FunBut — Lets a function key act like a rapid series of left mouse button events.
DC — A handy program for people who use an Amiga 1020 5 1/4 inch drive as an AmigaDOS floppy. A Workbench program that sends a DiskChange signal to the operating system: instead of typing "diskchange df2:" over and over again, just click on the icon. C source included.
System config — File makes screen 80 columns wide of text in the Scribble! word processor.
Dick2Ram — 2 programs to move the Scribble! spelling dictionary to and from the RAM disk.
Lexical — Analyzes a text file and gives the Gunning-Fog, Flesch, and Kincaid indices which measure readability.
HexDump — Modula-2 program to display memory locations in hexadecimal.
Tartan — AmigaBasic; design Tartan plaids.
DirMaster — Disk catalog program.
BMP — plays 8SVX sampled sounds in the background while something else is happening in the Amiga, as your Amiga is booting, for example.
ShowPt — CLI program changes your pointer to a given pointer.
AMICUS 26 also has a collection of mouse pointers, & Workbench program to display them



For PDS orders, please use form on page 96
Visa and MasterCard is available on orders of $20.00 or more.

The Fred Fish Collection

Due to the increasing size of the Fred Fish Collection, only the latest disks are represented here. For a complete list of all AC, AMICUS, and Fred Fish Disks, cataloged and cross-referenced for your convenience, please consult the current *AC's Guide To The Commodore Amiga* available at your local Amazing Dealer.

Fred Fish Disk 320

AmigaTrek A continuation of Mike's Amiga Trek stories, which are parodies of the Star Trek series, with an Amiga flavor. Earlier stories are on disk 278. Author: Mike Smithwick

AmiOmega Amiga port of the Omega game. Omega is similar to hack or rogue, but is much more complex. There is a city, several towns, a wilderness, lots of dungeons, a multitude of monsters, lots of spells, magic items, etc. There are several quests to complete. All in all, it is an excellent game. Requires 1Mb or more of memory. Amiga version 1.0, binary only. Author: Laurence Brothers, Amiga port by Rick Golembiewski

Fred Fish Disk 321

DezHexBin An intuition based programmers tool to convert integers between decimal, hexadecimal, and binary. Very small. Version 1.1, includes source in assembly code. Author: Michael Djavidan

IconJ IconJ significantly enhances the IconX program, and is 100% compatible. It allows scripts to be executed by double-clicking the script's icon. Abilities include joining the script with the icon file itself, or calling it from any directory or disk, executing either AmigaDOS or ARexx scripts, outputing to any file or device, running interactive scripts and scripts that contain conditionals, and creating relative console windows. Includes a utility called AtatJ which attaches or detaches a script to/from an icon file. Version 1.0, includes source in JForth. By: Rich Franzen

Ifs An Iterated Function System viewer which graphically displays iterated function systems and allows the user to interactively create the affine functions that define such systems. An IFS can represent complex pictures very compactly. Simple IFSs can describe an infinite number of different and interesting fractal displays. Includes a number of displays that the author and others have discovered. Version 1.5, includes source in C. Author: Glen Fullmer

Planets Some routines ported to the Amiga by Bob Leivian, that compute the location of the planets (as viewed from a specific point on the earth) and the phase of the moon, for an arbitrary date and time. Includes source. Author: Keith Brandt VIII, Jim Cobb, F. T. Mendenhall, Alan Paeth, Petri Launiainen, Bob Leivian

Turtle A shared library of "turtle" functions for drawing in a RastPort. Includes source in assembly and C. Author: Thomas Albers

UnixDirs A program which intercepts calls to dos.library to add the UNIX style '.' and '..' syntax for current and parent directories, respectively, to file and path names. I.E., you can refer to files in the current directory as './foo' and files in the parent directory as '../foo', or any combination of the two. Includes source in assembly. Author: Murray Bennett and Mark Cyster

Whereis Another "find-that-file" utility. Whereis searches on your (hard-)disk for a file(name) and displays the path to that file. Some features are case independent search, wildcards, interactive mode (cd implemented), can display size and date of files, always abortable, can archive filenames for "ZOO" (like fnams/recurdir), and no recursive procedures. Includes source in C. Version 1.18 (2-15-90). Author: Roland Bless

Fred Fish Disk 322

Gwin This is version 1.0 of GWIN. GWIN or Graphics WINdow is an integrated collection of graphics routines callable from C. These routines make it easy to create sophisticated graphics programs in the C environment. One-line calls give you a custom screen (ten types available), menu items, requestors, text, circles, polygons, etc. GWIN is a two-dimensional floating point graphics system with conversion between world and screen coordinates. GWIN includes built-in clipping that may be turned off for speed. Use of color and XOR operations are greatly simplified. Many examples of the use of GWIN are included in an examples directory. Examples include line/bar graph program, geographic mapping program, SPICE 2G.6 graphics post-processor, and others. Extensive documentation is included. Author: Howard C. Anderson.

Fred Fish Disk 323

ColorTools Three tools that manipulate the colors of your screen. Binary only. Author: Dieter Bruns

CZEd A complete midi package for use with all Casio CZ synthesizers. Contains a full fledged sound editor, a split simulator for CZ-101/1000/230S, a bank loader and a memory dump for CZ-1. This is a formerly commercial package now released as shareware. Binary only. Author: Oliver Wagner

LinkSound Two examples of functions that you can link with your own code to produce a short musical "beep" or a sound that is similar to striking a drum. Includes source. Author: Dieter Bruns

Show A very versatile program to display IFF ILBM files. Features realtime unpacking scroll, smart analysis of any IFF file, total control over display modes, simple slideshow processing, pattern matching, and a dozen other options. Only 9K. Version 2.0, binary only. Author: Sebastiano Vigna

Fred Fish Disk 324 A

This Fred Fish Disk is offered as an abridged disk until Fred can createa replacement disk. One program has been removed from this disk due to copyright problems.

ANSIEd Demo version of an ANSI screen file editor. It allows you to easily create and modify a screen of ANSI-style text/graphics on the Amiga. The standard ANSI color set (red, green, yellow, blue, magenta, cyan, white) and text styles (plain, boldface, underlined, italic) are provided, along with some simple editing and drawing functions. This demo version has the save features disabled. This is version 1.3.0, an update to version 1.2.0aD on disk 221. Binary only. Author: Greg Epley

DiskFree An small iconifiable intuition program that shows the amount of free space available on all mounted disk devices, both numerically and graphically. Version 1.0, shareware, binary only. Author: Dieter Kuntz

DPFFT An enhanced version of DPlot from disk 290. DPlot is a simple display program for experimental data, with the goals of supporting paging through lots of data and providing comfortable scaling and presentation. The enhancements for DPFFT include addition of a Fast Fourier Transform (FFT), display of a customized amplitude and phase spectrum, a prewhitening capability, and a Welch window for spectral smoothing. This is version 2.1, binary only. Author: A. A. Walma

Mailchk A mail client for Dnet, which will inform you of any new mail and will give the choice of viewing, deleting, or printing a message. Version 2.01, includes source. Author: Stephane Laroche

Fred Fish Disk 325

Batchman A program that allows the user to execute CLI programs and batch files simply by clicking on a gadget. It can be used as the center of a turnkey system, where the user simply clicks on gadgets to launch applications. Version 1.1, includes source in Modula-II. By: Michal Todorovic

DClock A "Dumb Clock" utility that displays the date and time in the Workbench screen title bar. Uses only about 2 percent of the CPU time and about 10Kb of memory. Also has an alarm clock feature and audible beep for programs that call DisplayBeep. This is version 1.12, an update to version 1.5 on disk 296, with many enhancements and a few bug fixes. Includes source. Author: Olaf Barthel

DoRevision This program implements easy creation of source code revision headers (very similar to the log headers to be found at the top of the Amiga 'C' include files). Version 1.0, includes source. Author: Olaf Barthel

FAM A File Access Manager for the Amiga that allows multiple ARexx programs to access a buffered version of a directory in a consistent and serialized manner. It buffers all the names, dates, sizes and so on, for quick access. Version 1.1 with source. By: Darren New

FarPrint Debugging functions for programs which don't have any links to their environment. FarPrint consists of two major parts; a harbour process open to receive and distribute messages and requests, and a set of C functions to be linked into any program wishing to communicate with the FarPrint main process. This is version 1.5, an update to version 1.3 on disk 281, and adds a shared library as well as linker libraries for both Lattice and Aztec C. Includes source. Author: Olaf Barthel

KeyMacro A keyboard macro program, configurable via a text file, that also supports hotkey program execution. You can map up to eight functions to each key, including keys such as cursor keys, the return key, etc. Version 1.0, includes source. Author: Olaf Barthel

LifeCycles Some sort of biorhythm type program. No docs included. Version 2.0, binary only. By: Michal Todorovic

MemGuard MemGuard is a MemWatch-like program which has been rewritten in assembly language for maximum speed and efficiency. Unlike MemWatch MemGuard does not run as Task in a dummy loop but rather as a low-level interrupt routine which is capable of trapping memory thrashing even before exec might know of it and even while task switching is forbidden. In fact the low-memory area is checked each frame. Virtually no processing time is wasted, the interrupt routine does the check in about half a raster scan line's time. This program was contributed by Ralf Thanner, who spent three weeks programming & debugging it. In this program Ralf uses some very delicate tricks to let his interrupt routine work with Intuition alerts. Version III, binary only. Author: Ralf Thanner

RexxHostLib This is a shared library package to simplify the ARexx host creation/management procedure. Rexx-message parsing is also included making it possible to control ARexx from programs such as AmigaBASIC (can you imagine AmigaBASIC controlling AmigaTeX?). Includes source. Author: Olaf Barthel

Fred Fish Disk 326

CBDump This is a CLI utility for those who are working with the Amiga's clipboard device. It's sole purpose in life is to dump the current contents of the clipboard to stdout or by redirection to a pipe or a file. Useful for testing and interfacing with programs that do not support the clipboard. Source included. By: Stephen Vermeulen.

DispMod One of the series of ROBBS (Rexx Object Building Block System) modules by Larry Phillips. DispMod is a display module that only understands ARexx messages. It allows, under program control, the display of text and the acceptance of keyboard data. Version 0.11, includes source. Author: Larry Phillips

Itb This program converts an icon to an IFF picture (brush) file. It handles both single and alternate image (animated) icons. This is version 1.10 which adds a colour palette to the previous version from disk 85. Version 1.10, binary only. Author: Stephen Vermeulen.

MicroTerm A very small, very simple, almost brain-dead terminal program. Primarily useful as an example of how to talk to the console and serial devices. Version 0.1, includes source. Author: Stephen Vermeulen.

NeuronalNets Programs for playing with Neuronal Nets using Hopfield and Hamming algorithms. Binary. By: Uwe Schaefer

PopScreen A small hack to pop a hidden screen to the front from the CLI. This was written to allow the author to use VLTjr with other programs that also use custom screens. Source included. Author: Stephen Vermeulen.

Snap A tool for clipping text or graphics from the screen, using the clipboard device. Snap finds out character coordinates automatically, handles different fonts, keymaps, accented characters, and more. V1.4, an update to FF274. Includes source. By: Mikael Karlsson

VSnap This is an enhanced version of Snap 1.3, submitted by Steve Vermeulen, which adds the ability to save clipped graphics as IFF FORM ILBM's to the clipboard, so they can be imported to other programs that understand IFF and the clipboard. Dubbed it VSnap, since the official 1.4 Snap is also included on this disk. Includes source. By: Mikael Karlsson, enhancements by Steve Vermeulen

Fred Fish Disk 327

ARTM ARTM (Amiga Real Time Monitor) displays and controls system activity such as tasks, windows, libraries, devices, resources, ports, residents, interrupts, vectors, memory, mounts, assigns, fonts and hardware. Includes both a PAL and an NTSC version. This is version 1.0, an update to version 0.9 on disk 277. Binary only. Author: Dietmar Jansen and F. J. Mertens

MM An implementation of the game Mastermind. In this game you must try to guess a color combination which the amiga sets via a random generator. There are 6 colors which can be set in any combination. Includes source. Author: Dietmar Jansen

MRBackUp A hard disk backup utility that does a file by file copy to standard AmigaDOS floppy disks. Includes an intuition interface and file compression. This is version 3.4, an update to version 3.3e on disk 279. Binary only. Author: Mark Rinfret

Msh An Amiga file system handler that handles MSDOS formatted diskettes. You can use files on such disks in almost exactly the same way as you use files on native AmigaDOS disks. This is a fully functional, read/write version, that supports 8, 9, or 10 sector disks of 80 tracks, and should also work on 40 track drives and hard disks with 12 or 16 bit FAT of any dimension the FAT allows. Includes source. Author: Olaf Seibert

Softfont Converts portrait soft fonts for HP LaserJet compatible laser printers to landscape format. Includes source. Author: Thomas Lynch

Fred Fish Disk 328

AnalysiCalc A full featured system for numerical analysis and reporting. Includes a spreadsheet, graphics programs, documents and facilities for performing many commonly needed functions. Features include an 18000 by 18000 cell spreadsheet using virtual memory, random access to other saved spreadsheet formulas or values, easy save or merge of partial sheets, up to 400 windows on screen, ability to drive any cell from external macros, built in matrix algebra, random number generation, date arithemetic, and much more. This is version V24-01a, an update to FF176. Binary only. Author: Glenn Everhart

Hames Some miscellaneous programs from Chris Hames. DirWork V1.01 is a fast, small, simple efficient DirUtility. FSDirs V1.3 is a floppy accelerator program. VMK V27 is a small virus detector/killer that knows about 27 different viruses and can detect new ones. NoInfo V1.0 stops programs from producing *.info" files. Binaries only. Author: Chris Hames

RoadRoute A trip planner that takes a list of cities and a list of known routes between cities, and generates the distance and time required to reach your destination. An update to FF 251, with an expanded datebase of cities and roads for New Mexico, Texas, Oklahoma, Kansas, Nebraska, South Dakota, Louisiana, Arkansas, Missouri, Colorado and Mississippi, added by Fred Mayes and Gary Delzer. Includes source. By: Jim Butterfield, Fred Mayes, Gary Delzer

Fred Fish Disk 329

CPU Two programs, one in C and one in assembler, which check for CPU type. This version can detect 68000, 68010, 68020, and 68881 processors. Includes source. Author: Ethan Dicks, based on WhatCPU by Dave Haynie

DiskSpeed A disk speed testing program specifically designed to give the most accurate results of the true disk perform- ance of the disk under test. Automatically updates and maintains an ASCII database of disk results for tested disks. This is version 3.1, an update to FF288, with some source code cleanups and stress tests for CPU and DMA. Includes source in C. Author: Michael Sinz

Empire A complete rewrite, from the ground up, in Draco, of Peter Langston's Empire game. Empire is a multiplayer game of exploration, economics, war, etc, which can last a couple of months. Can be played either on the local keyboard or remotely through a modem. This is version 1.33w, an update to FF118, and includes many changes and enhancements. Binary . By: Chris Gray, David Wright, Peter Langston

FileSystems Displays AmigaDOS disk devices with information about the head geometry, BufMemType, and the lower level exec device. Includes source. Author: Ethan Dicks

OnePlane Removes the highest number bitplane from the WorkBench screen. Normally used to take Workbench screen from 2 bitplanes to 1 bitplane. This allows CON: style devices to scroll text faster. Includes source. Author: Ethan Dicks

Fred Fish Disk 330

Mostra A very versatile program to display IFF ILBM files. Features realtime unpacking scroll, smart analysis of any IFF file, total control over display modes, simple slideshow processing, pattern matching, and a dozen other options. Only 14K. This is version 1.0, an upgrade to the Show program on disk 323, and adds SHAM, double buffering, faster decompression, color cycling, TeXdocs, startup files for easy customizing, and complete WorkBench support through ToolTypes and Style icons. Binary only. By: Sebastiano Vigna

Palette A tool which allows you to change another program's custom screen colors. This is version 1.1, an update to the version on disk 55. New features include checks for WorkBench startups, checks for HAM, Half Brite, or more than five bitplanes, and more graceful exits. Includes source in assembly. Author: Randy Jouett, CJ Fruge, Carolyn Scheppner, Charlie Heath

Vt100 A vt100 emulator for the Amiga, which also supports various file transfer protocols like kermit, xmodem, ymodem, zmodem, etc, has an Arexx port, can use custom external protocol modules, and more. This is version 2.9a, an update to version 2.9 on disk 275. Includes source. Author: Dave Wecker, Tony Sumrall, Frank Anthes, and Chuck Forsberg

XprKermit An Amiga shared library which provides Kermit file transfer capability to any XPR-compatible communications program. Supports version 2.0 of the XPR Protocol specification. Version 1.5, includes source. Author: Marco Papa, Stephen Walton

Fred Fish Disk 331

CRobots A game based on computer programming. Unlike arcade type games which require human input controlling some object, all strategy in CRobots is condensed into a C language program that you design and write, to control a robot whose mission is to seek out, track, and destroy other robots, running different programs. All robots are equally equipped, and up to four may compete at once.Version 2.2w, an update to FF311. Binary only, source available from author. By: Tom Poindexter, Amiga version by David Wright

Csh Version 4.01a of a csh like shell derived from Matt Dillon's shell, version 2.07. This is an update to version 4.00a on disk 309. Changes include mostly bug fixes and corrections. Includes source. Author: Matt Dillon, Steve Drew, Carlo Borreo, Cesare Dieni

For PDS orders, please use form on page 96
Visa and MasterCard is available on orders of $20.00 or more.

Iff2Ex A program to convert IFF pictures to an executable. It can handle NTSC/PAL, interlace and overscan. Version 1.0, binary only. Author: Pieter van Leuven

LhArcA An intuitionized and faster version of lharc for the Amiga. Requires ARP library. Version 0.99a, binary only. Author: Haruyasu Yoshizaki, Amiga version by Stefan Boberg

LVR Link Virus Remover. A program that recursively searches directories for link viruses in executable files. This is version 1.20, binary only. By: Pieter van Leuven

NTSC-PAL Utilities which allow Amigas with the new ECS 1Mb Agnus to easily switch between PAL and NTSC display modes. Version 1.0, includes source in assembly. Author: Nico Francois

PatchLoadSeg This program patches the loadseg routine to automatically detect link viruses when a program is loaded. Displays an alert when a virus is detected in a program being loaded for execution. Version 1.20, includes source. Author: Pieter van Leuven

VirusUtils Two programs to detect viruses on disk and in memory. Virushunter removes all known viruses in memory. Viruskiller removes all known viruses in memory and after removing the viruses the disks can be checked without the virus copying itself to the disks. Version 3.60, binary only. Author: Pieter van Leuven

Fred Fish Disk 332

AniPtrs Some cute animated pointers. I have adopted one of them as my permanent replacement for the boring red arrow. Binary only. Author: Bob McKain

DevPatch A program that installs a patch for OpenWindow to check the NewWindow structure. If the title matches a specific string, the height will be forced to 45 pixels. This helps to reduce chip memory usage for programs that open overly large windows and then seldom use them. Includes source. Author: Jorrit Tyberghein, Nico Francois, P. Marivoet

Helper A little InputEvent hack, activated via the HELP key. Originally meant to provide a unique method of giving the user help (you don't have to put that help stuff into your own program). Now also contains a color requester and a small notepad. Version 1.01, includes source. Author: Michael Balzer

K1_Editor An editor for the Kawai K1(m) synthesizer with two auxiliary programs for managing sound dumps. This is version 1.00, shareware, includes source. Author: Michael Balzer

Kryptor A small, simple and comfortable file encoder/decoder. Version 1.0, includes source. Author: Michael Balzer

RevBut Another InputEvent hack, giving you a toggling right mouse button. Version 1.0, includes source. Author: Michael Balzer

Fred Fish Disk 333

MultiPlot A package for making 2D plots conveniently. Tim Mooney wrote the original program, which was then enhanced by Alan Baxter with a nicer user interface, support for the PLT: device, and support for file conversions. Rich Champeaux and Jim Miller wrote the PLT: handler which emulates a plotter by accepting HP-GL commands, creating a raster image, then dumping it to any preferences supported graphics printer. This is version XLNb, an update to FF292, and includes many bug fixes, style changes, and enhancements. Includes source. Author: Alan Baxter, Tim Mooney, Rich Campeaux, Jim Miller

Fred Fish Disk 334

FBM An Amiga port of the Fuzzy PixMap image manipulation library. This package allows manipulation and conversion of a variety of color and B&W image formats. Supported formats include Sun rasterfiles, GIF, IFF, PCX, PBM bitmaps, "face" files, and FBM files. Also has input converters for raw images, like DigiView files, and output converters for PostScript and Diablo graphics. Besides doing format conversion, some of the other image manipulation operations supported include rectangular extraction, density and contrast changes, rotation, quantization, halftone grayscaling, edge sharpening, and histograms. Version 0.9, binary only. Author: Michael Mauldin; Amiga port by Kenn Barry

PPMore A "more" replacement program that reads normal ascii text files as well as files crunched with PowerPacker. The crunched files can result in consider disk space savings. Version 1.5, binary only. Author: Nico Francois

PPShow A "show" program for normal IFF ILBM files or ILBM files crunched with PowerPacker. The decrunching is done auto- matically as the file is read. Version 1.0, binary only. Author: Nico Francois

WhatIs A neat little utility which not only recognizes a wide variety of file types (executables, IFF, icons, zoo files, etc), but prints interesting information about the structure or contents of the recognized file types. Version 1.2a, binary only. Author: J. Tyberghein

Fred Fish Disk 335

BoingDemo Demo version of a neat game due for release in March 1990. It is fully functional but the play time is limited to five minutes per play. Version 0.30, binary only. Author: Kevin Kelm, Alternate Realities

DTC A utility providing a simple calendar which can hold and show appointments. It may be useful in managing your time. Its chief goals were to provide day, week and month at a glance for any date between 1/1/0001 and 12/31/9999, defaulting to the current date. It is menu driven and fairly easy to use. Includes source in Fortran. Author: Mitch Wyle, Amiga port by Glenn Everhart

SeeHear A program to do a spectrogram of a sampled sound file. This is a graph with time on one axis, frequency on the other and the sound intensity at each point determining the pixel color. With source in C, including FFT routine. This is version 1.1. Author: Daniel T. Johnson

Fred Fish Disk 336

Car A two-dimensional full screen scrolling racing game with realistic four channel stereo sound and overscan. for either NTSC or PAL Amigas. The goal is to guide your car around one of ten selected tracks. Each track has its individual high score list. Version 2.0, binary only. Author: Anders Bjerin

FileWindow A completely public domain file requester which may be used in any program, even commercial ones. It uses dynamically allocated memory to hold the file names so the only limitation is the amount of memory available. Includes a filter option to limit display of filenames to only ones with a specific extension. Names are automatically sorted while they are being read and displayed. V1.10, includes source. By: Anders Bjerin

MiniBlast A shoot'em up game which runs just fine in a multi-tasking environment. At last you can enjoy a satisfying megablast while you are writing a boring essay. Shoot anything that moves, and if it doesn't move, shoot it anyway. V1.00, binary only. By: Anders Bjerin

Sys A game built on the addictive game PONGO but with several added features. You have been assigned the demanding task of cleaning viruses from your SYSOP's hard disk. To kill a virus, you simply kick a disk at it. There are fifty different levels, and on each level, the speed will increase and the viruses will be smarter and start to hunt you. V2.10, binary only. By: Anders Bjerin

Fred Fish Disk 337

CManual A complete C manual for the Amiga which describes how to open and work with screens, windows, graphics, gadgets, requesters, alerts, menus, IDCMP, sprites, etc. The manual consists of more than 200 pages in 11 chapters, together with more than 70 fully executable examples with source code. When unpacked, the manual and examples nearly fill up three standard Amiga floppies. This is version 1.00 and includes source for all examples. Author: Anders Bjerin

Fred Fish Disk 338

Cpp This is a copy of the Decus cpp, ported to the Amiga. This cpp is more powerful and complete than either of the built in cpp's in Manx or Lattice C. This is an update to the version on disk 28. It has had some ANSI features added. Includes source. By: Martin Minow, Olaf Seibert

SASTools Various submissions from "Sick Amiga Soft". Includes some virus tools, some screen hacks, some small games, and miscellaneous utilities. Includes source in assembly and Modula-II. Author: Jorg Sixt

SID A very comprehensive directory utility for the Amiga that supports at least a couple of dozen different commands for operating on files. Version 1.06, binary only. Author: Timm Martin

Fred Fish Disk 339

PCQ A freely redistributable, self compiling, Pascal compiler for the Amiga. The only major feature of Pascal that is not implemented is sets. This is version 1.1c, an update to version 1.0 on disk 183. It is much enhanced and about four times faster. Includes the compiler source and example programs. Author: Patrick Quaid

Fred Fish Disk 340

NorthC A complete freely redistributable C environment for the Amiga based on the Sozobon Ltd C compiler, Charlie Gibb's assembler, the Software Distillery's linker, and portions from other sources. Steve has pulled everything together and added some enhancements in the process. Version 1.0, partial source only. By: Steve Hawtin, et. al.

Plplot A library of C functions useful for scientific plotting on the Amiga. The library is Lattice C compatible. Contour plotting, three dimensional plotting, axis redefinition, log-log plotting and multiple subpages are a few of Plplot's features. The plots can be displayed on a monitor or sent to a graphics file for subsequent printing. This is version 2.6, and update to version 1.00 on FF222. This version includes a greatly improved intuition interface, preferences support for hardcopy, several new device drivers, and the capability of adding additional device drivers easily. Includes source. Author: Tony Richardson

SpeakerSim Demo version of SpeakerSim 2.0, a loudspeaker CAD program. Simulates vented (Thiele-Small) and closed box systems. Also simulates 1st, 2nd, and 3rd order high and low pass filters. Binary only. By: Dissidents

Fred Fish Disk 341

P2C P2C is a tool for translating Pascal programs into C. It handles the following Pascal dialects: HP Pascal, Turbo/UCSD Pascal, DEC VAX Pascal, Oregon Software Pascal/2, Macintosh Programmer's Workshop Pascal, Sun/Berkeley Pascal. Modula-2 syntax is also supported. Most reasonable Pascal programs are converted into fully functional C which will compile and run with no further modifications. V1.13 Includes source. Author: Dave Gillespie, Amiga port by G. R. (Fred) Walter

Fred Fish Disk 342

IE This is an icon editor which can create and modify icons up to 640x200 pixels in size (also dual render). It can set stack size, position of icon (also free-floating), default tool, 10 tool types and control over opened window. It can also generate the C source code behind the icon for program inclusion. Version 1.0, binary only, source available from author. Author: Peter Kiem

SKsh A ksh-like shell for the Amiga. Some of its features include command substitution, shell functions with parameters, aliases, local variables, local functions, local aliases, powerful control structures and tests, emacs style line editing and history functions, I/O redirection, pipes, large variety of built-in commands, Unix style wildcards, Unix style filename conventions, filename completion, and coexistence with scripts from other shells. Very well documented. Version 1.4, an update to version 1.3 on disk 309. New features include a "tiny" version, a working case construct, support for resident commands, smaller and faster external commands, and more. Binary only. Author: Steve Koren

Softfont Converts portrait soft fonts for HP LaserJet compatible laser printers to landscape format. This is an update to FF327. Includes source. Author: Thomas Lynch

Fred Fish Disk 343

SnakePit A simple, yet addictive game in which you must get the snake (you) off of the screen. There are, however, some rough spots and some obstacles that may need to be overcome. Excellent example of a game that is as system friendly as possible(with source). By: Michael Sinz

SoftSpan Soft Span BBS program. Intuitive, command-line based menu system with message bases, up/down loads, file credit system, extensive help system, etc. This is shareware version 1.0, binary only, lattice C source code available from the author. Author: Mark Wolfskehl

StockBroker A program that helps you follow the recent table of exchange from one (or more) share(s). But of course you must tell the Amiga the recent table of exchange every day. Requires AmigaBASIC. Binary only. Author: Michael Hanelt

Fred Fish Disk 344

Keyboard Functions to translate RAWKEY Intuition messages into usable keycodes. Translation into Modula-2 of C source (by Fabbian G. Dufoe, III) on disk 291. Version 1.0. Includes source. Author: Fabbian G. Dufoe III, Peter Graham Evans

RKMCompanion A two disk set of material created by Commodore for use with the 1.3 revison of the Amiga ROM Kernel Reference Manual, Libraries and Devices, published by Addison-Wesley. Almost 300 files, including C source code examples and executables, have been packed into two lharc archives, one for each disk of the two disk set. These examples are not public domain, but may be used and distributed under the conditions specified in the copyrights. Author: Commodore Business Machines, Inc.

Fred Fish Disk 345

CRobots A game based on computer programming. Unlike arcade type games which require human input controlling some object, all strategy in CRobots is condensed into a C language program that you design and write, to control a robot whose mission is to seek out, track, and destroy other robots, each running different programs. All robots are equally equipped, and up to four may compete at once. This is version 2.3w, an update to FF331. Binary only, source available from author. Author: Tom Poindexter, Amiga version by David Wright

Du Prints number of disc blocks used in selected files or directories. Modified from original version on disk 48 to make output more readable, and handle *C exit. Includes source. By: Joe Mueller, enhancements by Gary Duncan

GetImage An enhanced version of "gi" from disk 14. It now looks for the GRAB marker, in the brush file, instead of assuming that it is at a specific place, sets up the PlanePick value in the Image structure, and deletes any unused bitplanes to save memory and disk space. Includes source. Author: Mike Farren, enhancements by Chuck Brand

MemFrag Displays number of memory chunks/sizes to show memory fragmentation. Chunks are displayed as 2**N bytes which is a rough guide but still useful. This is an enhanced version of "Frags" from disk 69. Includes source. By: Mike Meyer, enhancements by Gary Duncan

Roses A program that draws sine roses. Implements an algorithm given in the article "A Rose is a Rose ..." by Peter M. Maurer in American Mathematical Monthly, Vol 94, No. 7, 1987, p 631. A sine rose is a graph of the polar equation "r = sin(n*d)" for various values of n and d. Author: Carmen Artino

Unshar This program extracts files from Unix shar archives. It scores over similar programs by being small and fast, handling extraction of subdirectories, recognising a wide variety of 'sed' and 'cat' shar formats, and handling large files spread across several shar files. This is version 1.3, an update to the version on disk 287. Includes C source. Author: Eddy Carroll

VcEd A Voice (Tone) Editor for the Yamaha 4 Operator series synthesizers. Binary only, source available from author. Author: Chuck Brand

X2X Cross converts between Motorola/Intel/Tektronix ASCII-hex files. These files are typically used for down-line-loading into EPROMS, or for transmission where binary files cause chaos. Handles S1, S2, S3, INTEL (inc USBA records), Tektronix (inc extended). Source included. Author: Gary Duncan.

Fred Fish Disk 346

Az A nice little text editor that is fast, simple to use, and very Amiga'ized. This is version 1.50, an update to FF 228, with lots of new features, bug fixes, and other improvements. Binary only. By: Jean-Michel Forgeas

CassEti Cassette tape label printer. Includes source in GFA Basic. Author: Thorsten Ludwig

FME Patch to AllocMem() to allow badly designed programs which request fast mem without necessity to be run on 512k machines. Includes source in assembler. Author: Holger Lubitz

GoWB Very small (296 bytes) and effective replacement for the well known "LoadWB" and "EndCLI" command pair. This release fixes a severe bug in the first version which used to guru if run out of a script. Includes source in C. Author: Oliver Wagner

PacketSupport A link library, for use with Lattice C, providing a few functions to handle DOS packet postage. Includes source. Author: Oliver Wagner

PatchNTSC OS fix to allow the growing number of PAL display programs to be run on NTSC machines. Will patch the Intuition OpenScreen() function to assure screens with PAL height to be opened in interlace mode. Includes source in assembler. Author: Oliver Wagner

TextPaint Second major release of the Ansi editor. All major bugs have been fixed, and a bunch of new options have been added, e.g. possibility to reload ansi files or CLI modules, 4 color option, optimized keyboard layout, new drawing modes, right mouse button support (like DeluxePaint) and much more. Binary only, shareware. By: Oliver Wagner

Timetest Working example to show the time() and gmtime() functions of the Lattice C support library. Includes source in C. Author: Oliver Wagner

WBD Possibly the smallest utility to set the workbench screen to any depth. Includes source in C. By: Oliver Wagner

Fred Fish Disk 347

Cursor A 3-pass BASIC Compiler for BASIC programs written in AmigaBASIC, does not yet support all of the BASIC commands but is able to compile itself. This is version 1.0, includes source. Author: Jurgen Forster

Drip Drip is an arcade style game with 15 floors (levels). You must move along the pipes of each floor and rust them to advance to the next level. Every 3 floors completed will entitle you to a bonus round where extra drips can be won. An extra drip will also be awarded for every 10,000 points. Binary only. Author: Art Skiles

Fred Fish Disk 348

ColorReq Describes the update to the color.library and has an example program, with source, that demonstrates its use. Author: Dissidents Software

DisEditor This is a demo of the dissidents shareware text editor. Version 1.1, binary only. Author: Dissidents Software

DisSecretary This program can be used to file information in a "file cabinet" type environment. It is well suited for jobs such as maintaining a disk catalog, or user group membership, etc. Included is a data file of the library catalog, disks 1 to 310. Version "Wanda", binary only. Author: Dissidents Software

FileIO Contains updated files for version 1.6 of the dissidents requester library. There is a bug fix to the library as well as a new function. See FF257 for the complete documentation, and examples. By: Dissidents Software

ILBMLib Contains updated files for the dissidents ilbm.library on FF237, with new lib features and a new library. Also included is a much improved (better organized) doc file, and new C examples that show how to use the library for any kind of IFF file. See FF237 for other examples. Author: Dissidents Software

InstallLibs A program to copy files to the LIBS: dir of a boot disk. Can be used to create a handy installation program (hard disks especially) for programs that need disk-based libraries. Includes source. By: Dissidents Software

SAMP An IFF sampled sound format designed for professional music use. It can be used for 16-bit samples, multiple waveforms, etc. Includes a SAMP reader/writer shared library, interface routines, and programming examples. Also includes a program to convert 8SVX to SAMP. Author: Dissidents Software

Fred Fish Disk 349

MED A music editor much like SoundTracker. A song consists of up to 50 blocks of music, which can be played in any order. Editing features include cut/paste/copy tracks or blocks, changing the vibrato, tempo, crescendo, and note volume. Other features include switching of the low-pass-filter on or off on a per song basis, and a cute little animated pointer of a guy doing "jumping jacks" in time to the music! Version 2.00, an update to version 1.12 on FF255. Now includes full source. Author: Teijo Kinnunen

Fred Fish Disk 350

Icons A large variety of icons for many uses, of practically every description. Most are animated. By: Bradley W. Schenck

MemMometer A program that opens a narrow window and graphically displays your memory usage like a gauge. Based on WFrags, by Tomas Rokicki. Version 2.10, includes source. Author: Howard Hull

Stitchery This shareware program loads in IFF images and creates charted patterns from them for use in counted cross-stitch and other forms of needlework. It requires one megabyte of memory to run, and works best with a good high-resolution printer for printing the patterns. The Stitchery was written with The Director and the Projector is included. Version 1.21. Author: Bradley W. Schenck

TrackUtils Two utilities that deal with disk tracks. TCopy copies one or more tracks from one disk to another, and is useful for copying part of a floppy disk into RAD: during bootup. TFile creates a dummy file which "marks" a specified range of tracks, preventing AmigaDOS from using them and allowing them to be used for raw trackdisk data. Includes C source. Author: Eddy Carroll

To Be Continued.....

In Conclusion

To the best of our knowledge, the materials in this library are freely distributable. This means they were either publicly posted and placed in the public domain by their authors, or they have restrictions published in their files to which we have adhered. If you become aware of any violation of the authors' wishes, please contact us by mail.

IMPORTANT NOTICE!

This list is compiled and published as a service to the Commodore Amiga community for informational purposes only. Its use is restricted to non-commercial groups only! Any duplication for commercial purposes is strictly forbidden. As a part of Amazing Computing™, this list is inherently copyrighted. Any infringement on this proprietary copyright without expressed written permission of the publishers will incur the full force of legal actions.

Any non-commercial Amiga user group wishing to duplicate this list should contact:

PiM Publications, Inc.
P.O.Box 869
Fall River, MA 02722

PiM Publications Inc. is extremely interested in helping any Amiga user groups in non-commercial support for the Amiga.

For PDS orders, please use form on page 96
Visa and MasterCard is available on orders of $20.00 or more.

*(**CES**, continued from page 79)*

Taito

Taito has four games scheduled for release in August, although we have no details yet. ***Castle Master***, ***Day of the Pharoh***, ***Kiwi Kraze***,and ***Operation Thunderbolt*** (an arcade hit) are all coming soon and will sell for $34.95 each. **Inquiry # 313**

Taito Software Inc.
267 West Esplanade
N. Vancouver, B.C.
Canada V7M 1A5
(604) 984-3344

Virgin Mastertronic

Spirit of Excalibur King Arthur has died, and the kingdom is in chaos. You select from a group of Knights of the Round Table those who can help you restore peace. The game has more than 2.5 megabytes of graphics, which should make this visually enjoyable. Due in August. $49.99 **Inquiry # 314**

If you enjoyed those crazy little spots in the 7-Up commercials, then ***SPOT - The computergame*** is for you. Promised to be the most addictive strategy game since Tetris, you can play alone or with three friends through nine levels and 512 pre-programmed playfields for hours. October. $39.99 **Inquiry # 315**

Alice in Wonderland features some amazingly sharp graphics, pop-up menus, on-screen maps, help, and icons for every object. October. $49.99. **Inquiry # 316**

Quasar- take by force an enemy planet at the far side of the galaxy. Complex strategy will be needed here as you conquer portions and establish bases of the lower worlds. Due in September. $49.99 **Inquiry # 317**

Monopoly due in August or thereabouts. You can play against a friend or the computer. $39.99 **Inquiry # 318**

Virgin Mastertronic
18001 Cowan St. Suites A & B
Irvine, CA 92714
(714) 833-8710
FAX(714) 833-8717

•AC•

*(**Accelerators**, continued from page 12)*

If you choose the Hard Drive option, all that will be required is a simple ROM installation, and mounting the 3.5-inch Quantum hard drive to the drive chassis. One thing to note is that the hard drive comes ready to install in one of the 3.5-inch drive bays. An optional adapter kit is available if you want to install the drive in the 5.25-inch bay. Formatting and software installation is simple. GVP includes a disk and automated software to format the disk. In both cases, if you read the rather small, but sufficient manual, you should complete the installation flawlessly. The result is a very fast Amiga, with a very fast hard drive—nice! You wouldn't believe how fast a cold boot is!

As for software compatibility, I have had no problems so far, except for a few games. Even then, you can turn off the accelerator by a jumper on the accelerator board, the drawback being that you have to open your Amiga to do that. Otherwise, the compatibility has been exceptional.

Overall the Impact boards are fantastic. I have only two minor complaints about them. First, I wish the manual was a little more substantial. The second complaint is not really toward GVP, but toward the FCC. The GVP line of accelerators has currently met an official FCC class A rating, while all the other accelerators currently have an FCC B rating. The GVP Accelerator line is pending official class B certification from the FCC, but this certification often takes much longer than anticipated (probably due to bureaucracy). Although all GVP products do meet FCC regulations, it's nice to have that official class B rating. It's all just a matter of semantics.

But, overall the Impact boards have worked fantastically, and we have been using an A3001/8MB with the 80MB Quantum drive daily for the last three months.

THE IMTRONICS HURRICANE BOARDS

The first thing about the Hurricane board: read the manual!. The manual is simple but informative, and a quick read is sure to be followed by a trouble-free installation. The installation is relatively straightforward. Just plug it in and go. There are no EPROMs to plug in, and there is only one jumper on the board (the switch from the 68000 mode to 68030 mode). However, this was the only accelerator which required some simple software installation to work properly. This is easily accomplished with the included software on the installation disk.

The built-in SCSI autobooting hard disk interface is very nice, and provides fast disk access with your choice of SCSI hard disk. SCSI hard disks are fast, but SCSI compatibility problems can make them difficult to integrate. To be safe you should contact the manufacturer to verify compatibility before spending a chunk of change on a hard drive. Although hard disk performance was not taken into consideration in our test, we have found that the Hurricane SCSI interface was one of the faster on the market boasting a 750KB/sec transfer rate and 315 seeks per second. The optional SCSI II controller could only be faster. Also, the SCSI controller has been designed to support other SCSI devices such as SCSI laser printers. (Users will be notified when this option becomes available.) However, the only problem I had with the Hurricane board stemmed from a minor SCSI interface conflict. Reading the manual and experimenting a bit quickly resolved this problem.

The Hurricane boards also have a zero-slot solution. The accelerator, SCSI interface, and the 2MB memory only take up the coprocessor slot (leaving all five Zorro slots open). The Hurricane boards also work with the Imtronics M2000 memory boards, bringing up to 16MB memory to the board.

Overall, the Imtronics Hurricane accelerators are good workhorse accelerators that have been around for a while. The only thing I personally don't like about the Hurricane accelerators is the physical design of the boards. To help bring the Hurricane boards up to an FCC B rating, the two boards that make up the Hurricane are situated so that the component sides face each other, without room for the best ventilation. Although the temperature doesn't even come to the top temperature specification of the chips on the boards, I believe a little thermodynamic engineering couldn't hurt. Aside from this tiny, and personal, gripe, the Hurricane board is a very good workhorse accelerator.

THE COMMODORE AMIGA A2630 BOARD

If there was one word to describe the A2630 board it would be "elegant". There are no fancy options, no hard disk controllers, just a 25 MHz 68030/68882 and 2 MB-32 bit DRAM. There isn't even any software that comes with the board. The simple 28-page manual (which includes schematics of the A2630) tells you to plug the A2630 into the coprocessor slot. That's it. It runs flawlessly.

The compatibility is fantastic, and to change into the 68000 mode, all you do is boot the Amiga while holding the two-mouse buttons down. A requestor will pop-up where you can choose either the 68000 mode, 68030 mode, or a future UNIX boot mode. Now that's elegant. The board is also FCC B compliant, which is evident by the shielding around the processors. Very nice.

There's only one problem with this accelerator. If you want to expand it to 4MB, you are going to have to solder the chips down yourself, or take it to a dealer. (It uses the same ZIP type DRAM that the Amiga 3000 uses.) Barring this detour, this is a perfect accelerator for someone who doesn't want/need/can't afford a hot rod hard-disk/screaming accelerator.

THE BOTTOM LINE

The bottom line is that if you want to buy an accelerator, you're not only going to buy a board that's going to make your computer faster, but you're going to extend your Amiga's performance—well into the workstation category. It's not going to be cheap, as many boards with accessories cost into the $3500+ category, but it's much cheaper (by thousands of dollars) than buying a Silicon Graphics color workstation. (Besides, you can't play F/A-18 Interceptor on a Silicon Graphics workstation.)

During the next few months we will present complete reviews and reports on these marvelous wonders that push the Amiga to its limits. We will examine accelerators in conjunction with ray tracing, disk access, modifications, options, etc. Until then, look out! Speed is addictive! •AC•

Products Mentioned:

Great Valley Products (GVP)
600 Clark Avenue
King of Prussia, PA 19406
(215) 337-8770
FAX (215) 337-9922
Inquiry # 331

Imtronics, Inc.
12301 S.W. 132 Court Street
Miami, FL 33186
(305) 255- 9302
FAX (305) 255-6903
Inquiry # 332

Commodore Business Machines
1200 Wilson Drive
West Chester, PA 19380
(215) 431-9100
Inquiry # 333

Amazing COMPUTING / AMIGA & AC's GUIDE / AMIGA

Name _____

Address _____

City _____ State _____ Zip _____

Charge my ☐ Visa ☐ MC # _____

Expiration Date _____ Signature _____

MasterCard® VISA®

All Charges are subject to a $20.00 minimum (charges under $20.00 will receive a $2.00 service charge).

Please circle if this is a **New Subscription** or a **Renewal**

PROPER ADDRESS REQUIRED. In order to expedite and guarantee your order, all large Public Domain Software orders, as well as most Back issue orders, are shipped by United Parcel Service. UPS requires that all packages be addressed to a street address for correct delivery.
PAYMENTS BY CHECK. All payments made by check or money order must be in US funds drawn on a US bank.

One Year Of Amazing	**Save over 49%** 12 monthly issues of the Number One resource to the Commodore Amiga, **Amazing Computing** at a savings of over $23.00 off the newsstand price!	☐ $24.00 U.S ☐ $44.00 Foreign Surface ☐ $34.00 Canada and Mexico
One Year of AC SuperSub!	**Save over 47%** 12 monthly issues of **Amazing Computing** PLUS AC's GUIDE / AMIGA 3 Product Guides a year! A savings of $32.25 off the newsstand price.	☐ $36.00 U.S ☐ $64.00 Foreign Surface ☐ $54.00 Canada and Mexico
Two Years Of Amazing	**Save over 59%** 24 monthly issues of the Number One resource to the Commodore Amiga, **Amazing Computing** at a savings of over $56.80 off the newsstand price!	☐ $38.00 US (sorry no foreign orders available at this frequency)
Two Years of AC SuperSub!	**Save over 56%** 24 monthly issues of **Amazing Computing** PLUS AC's GUIDE / AMIGA 6 Product Guides a year! A savings of $77.50 off the newsstand price.	☐ $59.00 US (sorry no foreign orders available at this frequency)

Please circle any additional choices below: *(Domestic and Foreign air mail rates available on request)*

Back Issues: $5.00 each US, $6.00 each Canada and Mexico, $7.00 each Foreign Surface.

V1.1 V1.2 V1.3 V1.4 V1.5 V1.6 V1.7 V1.8 V1.9 V2.1 V2.2 V2.3 V2.4 V2.5 V2.6 V2.7 V2.8
V2.9 V2.10 V2.11 V2.12 V3.1 V3.2 V3.3 V3.4 V3.5 V3.6 V3.7 V3.8 V3.9 V3.10 V3.11 V3.12 V4.1
V4.2 V4.3 V4.4 V4.5 V4.6 V4.7 V4.8 V4.9 V4.10 V4.11 V4.12 V5.1 V5.2 V5.3 V5.4 V5.5 V5.6

Back Issue Volumes: Volume 1-$19.95* Volume 2-$29.95* Volume 3-$29.95* Volume 4-$29.95*
*All volume orders must include postage and handling charges: $4.00 each US, $7.50 each set Canada and Mexico, and $10.00 each set for foreign surface orders. Airmail rates available.

Freely Distributable Software:

Subscriber Special (yes, even the new ones!)

1 to 9 disks	**$6.00 each**
10 to 49 disks	**$5.00 each**
50 to 100 disk	**$4.00 each**
100 or more disks	**$3.00 each**

$7.00 each for non subscribers **(three disk minimum on all foreign orders)**

Amazing on Disk:
AC#1. . .Source & Listings V3.8& V3.9
AC#2. . .Source & Listings V4.4
AC#3. . .Source & Listings V4.5 & V4.6
AC#4. . .Source & Listings V4.7 & V4.8
AC#5. . .Source & Listings V4.9
AC#6. . .Source & Listings V4.10 & V4.11
AC#7. . .Source & Listings V4.12 & V5.1
AC#8. . .Source & Listings V5.2 & 5.3
AC#9. . .Source & Listings V5.4 & V5.5
AC#10 ..Source & Listings V5.6 & 5.7

InNOCKulation Disk: IN#1...Virus protection

AMICUS

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26

Fred Fish Disks

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	NA	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
76	77	78	79	NA	81	82	83	84	85	86	NA	88	89	90	91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125
126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200
201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225
226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250
251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275
276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300
301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325
326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350

(NA Denotes disks removed from the collection)

Subscription: $_____

Back Issues: $_____

PDS Disks: $_____

Total: $_____

Please complete this form and mail with check, money order or credit card information to:

PiM Publications, Inc.
P.O. Box 869
Fall River, MA 02722-0869

Please allow 4 to 6 weeks for delivery

Complete Today, or Telephone 1-800-345-3360

PRESENTING

WORLD OF AMIGA IN CHICAGO

★ *Starring* ★

THE AMAZING AMIGA

★ *Featuring* ★

Amiga Hardware • Amiga Software
Amiga Accessories • Seminars • Bargains

Rosemont O'Hare Expo Center,
Rosemont, Illinois

October 5-7, 1990
Friday, Saturday & Sunday 10am–5pm

Pre-registration:
$8 per day or $20 for 3 days
Deadline for pre-registration Sept. 18

Registration at show:
$10 for 1 day
$25 for 3 days
Registration includes exhibits and seminars.

SAVE WITH PRE-REGISTRATION

Please register me for the 1990 World of Amiga in Chicago at the Special Pre-registration Rate.

NAME _____

COMPANY (if applicable) _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

☐ One-day registration ($8) ☐ Three-day registration Oct.5-7 ($20)

Make Check or Money Order payable to
The Hunter Group, 3380 Sheridan Drive, Suite 120, Amherst, NY 14226

WORLD OF AMIGA IN CHICAGO

FOR MORE INFORMATION
Call (416) 595-5906
Fax (416) 595-5093

Other upcoming events produced by The Hunter Group include **COMMODORE AMIGA USERS FAIR** in Valley Forge, PA, September 15 and 16, 1990, and **WORLD OF COMMODORE AMIGA** in Toronto, November 30 to December 2, 1990.

Circle 111 on Reader Service card.

Newsflash:

"usually IMTRONICS INC. make accelerators, today they make HISTORY!"

Imtronics INC. is making history by introducing the worlds fastest PC clocked at 50 MHz.

The HURRICANE 2800

brings ultimate performance to your Amiga 2000. The 68030 CPU is clocked at 28 MHz and now also at 50 MHz, with the 68882 FPU up to 33 MHz. Now including a standard SCSI autobooting FFS hard drive controller which works under both the 68030 and the 68000. The board is asynchronous and gen-lock compatible. The hardware is switchable between 68030 and 68000 operation. A performance increase of more than 1200% compared to a stock Amiga is possible with 28 MHz and even 2000% can be achieved with our 50 MHz design. Memory is expandable with our MEMORY board and the complete system fits into only one slot!

The M2000 memory board

can be used with the HURRICANE 2800 and the HURRICANE 2000 accelerators. Ultrafast 32-bit RAM multiplies the performance of the HURRICANE boards. Due to our innovative design, the RAM speed on our board rivals those of 'burst' mode designed boards.

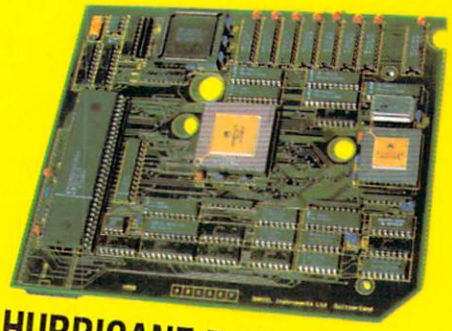
The HURRICANE 500

board turns your Amiga 500 into a 32-bit work station and is extremely easy to install in the 68000 socket. A performance of over 500% is possible with the 68020, additional performance increase can be reached with the 68881/68882 FPU of up to 33 MHz. The HURRICANE 500 is hardware switchable between the 68020 and the 68000 operation.

HURRICANE 2800

50 MHZ 28 MHZ

MEMORY BOARD



HURRICANE 500

IMTRONICS INC.

12301 South West 132 Court
Miami, Florida 33186

Phone: (305) 255 9302
Fax: (305) 255 69 03